Date of acceptance Grade

Instructor

Privacy-preserving proximity detection with secure multi-party computational geometry

Mandana Ghasemi

Helsinki August 19, 2019

UNIVERSITY OF HELSINKI
Department of Computer Science

HELSINGIN YLIOPISTO — HELSINGFORS UNIVERSITET — UNIVERSITY OF HELSINKI

Tiedekunta — Fakultet — Faculty	Laitos — Institution — Department
Faculty of Science	Department of Computer Science

Tekijä — Författare — Author
Mandana Ghasemi

Työn nimi — Arbetets titel — Title

Privacy-preserving proximity detection with secure multi-party computational geometry

Oppiaine — Läroämne — Subject
Computer Science

Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidoantal — Number of pages
	August 19, 2019	64 pages + 0 appendices

Tiivistelmä — Referat — Abstract

Over the last years, Location-Based Services (LBSs) have become popular due to the global use of smartphones and improvement in Global Positioning System (GPS) and other positioning methods. Location-based services employ users' location to offer relevant information to users or provide them with useful recommendations. Meanwhile, with the development of social applications, location-based social networking services (LBSNS) have attracted millions of users because the geographic position of users can be used to enhance the services provided by those social applications. Proximity detection, as one type of location-based function, makes LBSNS more flexible and notifies mobile users when they are in proximity. Despite all the desirable features that such applications provide, disclosing the exact location of individuals to a centralized server and/or their social friends might put users at risk of falling their information in wrong hands, since locations may disclose sensitive information about people including political and religious affiliations, lifestyle, health status, etc. Consequently, users might be unwilling to participate in such applications.

To this end, private proximity detection schemes enable two parties to check whether they are in close proximity while keeping their exact locations secret. In particular, running a private proximity detection protocol between two parties only results in a boolean value to the querier. Besides, it guarantees that no other information can be leaked to the participants regarding the other party's location. However, most proposed private proximity detection protocols enable users to choose only a simple geometric range on the map, such as a circle or a rectangle, in order to test for proximity.

In this thesis, we take inspiration from the field of Computational Geometry and develop two privacy-preserving proximity detection protocols that allow a mobile user to specify an arbitrary complex polygon on the map and check whether his/her friends are located therein. We also analyzed the efficiency of our solutions in terms of computational and communication costs. Our evaluation shows that compared to the similar earlier work, the proposed solution increases the computational efficiency by up to 50%, and reduces the communication overhead by up to 90%. Therefore, we have achieved a significant reduction of computational and communication complexity.

Avainsanat — Nyckelord — Keywords
location privacy, private proximity detection, point-inclusion problem, computational geometry

Säilytyspaikka — Förvaringsställe — Where deposited

Muita tietoja — övriga uppgifter — Additional information

Dedicated to my best teacher

Mr. Farzad Khalilimehr

for his encouragement, his advice, and his brilliant mindset.

Acknowledgement

Foremost, I would like to thank my supervisor Professor Valtteri Niemi and my advisor Kimmo Järvinen for the excellent academic mentorship and for having provided me with the opportunity to work under their supervision. I appreciate all their valuable suggestions and guidance of this research work, as well as to have allowed and trusted me to also follow my intuition during the research. Our many hours of discussions have taught me a lot.

I would also like to express my deep gratitude to Farzad Khalilimehr, who was there when I needed him the most. I appreciate his love, wisdom and unwavering support throughout the duration of my master's program. This accomplishment would not have been possible without him. Thank you Farzad for being the best friend and role model ever.

I cannot imagine having been able to get here without the support of many. I would like to thank Dr. Mohammad Mohammadi Aref and Dr. Pouyan Mohajerani for their help and valuable suggestions at the beginning of my studies. Their excellent academic experiences have lightened the way for me. Special thanks to my former colleagues, Dr. Milad Bahadori and Raine Nieminen for their scientific guidance and support. I further thank people in the Mathematics and Cryptography community of https://stackexchange.com/ who passionately share their knowledge with others on subjects that interest them.

I would like to offer my special thanks to all my friends at the University of Helsinki and Aalto University whom I met during these years. I was so lucky to meet such amazing and helpful people. Deepest appreciation for my dear friends Rita and Timo for their unconditional support and encouragement.

I warmly thank my dear uncle Ahmad Mohajerani, best trainer and life coach, who has opened my mind to different views. His valuable advice has always led me to success.

Last but not least, I owe my gratitude to all my family members, my parents and my awesome siblings Mehrnoosh, Alireza and Mehrangiz for their unconditional love and supporting me spiritually throughout my life. I would not be where I am today if it wasn't for them.

Contents

1 Introduction

In recent years, improvements in sensors and global positioning system (GPS) devices, as well as wireless communication technology, have resulted in a location-dependent information access paradigm, known as Location Based Services (LBS). Therefore, social networks, monitoring devices, and consumer-oriented applications can be enriched with locational data reported by users about where they are or how they are moving. These locational data can be employed to provide users with useful services such as *location-based queries*, *location-based social recommendations*, *traffic monitoring*, etc [Ghi13].

A *location-based social networking service (LBSNS)* combines social networking features with location-based services in which the geographical locations of mobile users are utilized to provide them with some notifications, e.g., informing users whether a social friend of them is in close proximity. This is called *proximity detection*.

However, the repeated release of information about the geographical location of mobile users raises serious concerns about the leaking and misusing of user location data. Since location data can be linked to a variety of information about individuals, collecting and analyzing location data would divulge some sensitive information about users including political views, sexual habits, religious affiliations, etc. To illustrate how the privacy of users might be violated by disclosing their exact location to an LBS, consider the following example.

Assume that Bob is looking for the nearest betting office using his mobile phone. A list of betting offices in his vicinity can be shown to him by a LBS in a publicly available web server (e.g., Google Maps, Mapquest). Since Bob is cautious about disclosing his gambling habits to Alice, he does not send his query directly to the LBS provider. Instead, he establishes an SSL connection with an *anonymizer*, which is a trusted server, in order to remove his user id from the query and to send it to the LBS. The answer is forwarded to Bob through the anonymizer. Here, the LBS provider needs the exact coordinates of the user in order to find his nearest betting office. Consider a scenario where Alice collaborates with the LBS and collects the query results along with the coordinates of users. It is not a difficult task for Alice to relate the coordinates to a specific user. For example, the location of users can be estimated by the phone companies within 50-300 meters (E911). Alice can triangulate Bob's mobile phone's signal to conclude which of the query results belongs to Bob. Other techniques include observing Bob in a specific place at the

query time or using publicly available databases for relating coordinates to users' address [KGMP07].

In practice, once users feel that by using a public LBS provider, their lifestyles or their political/religious affiliations are at risk of being disclosed to the wrong party, they might be reluctant to use such services. In addition, once the identity of the query source is revealed to the LBS, users may face unwanted advertisements, e-coupons, etc. Therefore, preserving location privacy is a fundamental requirement towards the prosperous deployment of location-based applications.

The techniques of *Secure Multi-party Computation* have brought many opportunities for cooperative computation, where untrusted companies or competitors are able to compute one function based on their own private inputs, ensuring that no party can derive more information except that what can be computed from a party's own input and the common output. The multi-party computation was first introduced by Yao [Yao82] and extended by Goldreich, Micali, and Wigderson [GMW87]. The suggested methodology in these works is to represent each functionality F as a Boolean circuit and then the participants run a protocol for every gate in the circuit. However, if the functionality F is complicated, their suggested solution will not be practical. This is the motivation for seeking special efficient solutions to specific cooperative computational problems.

1.1 Motivation

Secure multi-party computational geometry has got a wide range of applications in recent years. Its main goal is to design and implement privacy-preserving protocols for several geometric problems where each entity has its own private data. In [DA01b], the authors investigate a number of specific geometric problems including the *intersection* of two shapes, *range searching*, finding the *convex hulls*, *point-inclusion* problem, etc. In this work, we focus on the point-inclusion problem and design two privacy-preserving protocols that cover all arbitrary polygonal domains (except self-intersecting polygons). Here, we describe the problem with more detail and present some applications where a secure point-inclusion protocol is required.

Privacy-preserving point-inclusion problem: Alice has a polygon P and Bob has a private point M. They want to jointly figure out whether the point M lies inside the polygon P or not, without revealing any additional information to another party. Alice does not want to disclose any information about her polygon to Bob

(e.g, shape, size and its location), and likewise, Bob wants to hide the exact location of his point from Alice. Besides, Alice and Bob can only determine whether the point M is inside the polygon P or outside of it. Neither of them is allowed to learn the relative position between the polygon P and the point M, such as whether M is in the close distance of one of the polygon edges, or whether M is at the southeast side of P, and so on.

Applications:

The point-inclusion problem has a wide range of applications in *secure positioning* scenarios. For example, authors in [SS05] discuss the privacy concerns in pervasive sensor networks where users' voice, motion, or even body temperature are supervised. In such an environment, people want to know whether their current positions are being sensed, while the monitoring company does not want to reveal the sensing area. Using a privacy-preserving point-inclusion protocol, users can check whether they are located in the sensing area or not.

Another application of the point-inclusion problem is *proximity detection*, where a user can choose a geometric range on the map and query which friends of her/his are located inside the selected area. For example, Alice would like to enjoy her weekend at one religious festival held in the city center. She is interested to know whether her friend Catherine is also attending the festival in order to arrange some activities together. Using privacy-preserving proximity testing, she can determine if Catherine is located in that area before calling her. Another example can be as follows. Carol is a student who lands at an airport near her university. She is interested to check if one of her classmates is currently at the airport and can give her a ride to campus [NTL+11].

1.2 Contributions and outline

The main contributions of this thesis are as follows. We introduce an algorithm for solving the point-inclusion problem in complex polygonal domains without worrying about privacy concern (see Section 4.2). Then, we utilize the aforementioned algorithm to design and develop two privacy-preserving proximity detection protocols that allow one party to choose an arbitrary geographical area on the map and check whether his/her friends are located therein (see Section 4.3). We build our schemes based on public-key homomorphic encryption. Finally, we assess the security and performance aspects of the two suggested protocols. Our evaluation result shows

that the first solution outperforms a similar suggested protocol in terms of computational complexity and communication overhead. Whereas the second solution provides stronger privacy protection for both parties during the protocol execution.

The rest of this thesis is structured as follows: Chapter 2 discusses the background information. In Chapter 3 we explain the cryptographic and geometric primitives that are considered as building blocks for understanding the rest of our work. The system model and security requirements are formalized in Chapter 4, followed by an algorithm that solves the point-inclusion problem in a straightforward way without worrying about the privacy concern. Moreover, two new privacy-preserving protocols for the point-inclusion problem that covers all arbitrary polygonal domains are proposed in this chapter. Next, Chapter 5 evaluates the security and performance aspects of two proposed privacy-preserving schemes in the form of an analytical study. Finally, Chapter 6 presents the overall conclusions and suggestions for future work.

2 Background

This chapter studies the background information. The concept of privacy is discussed in Section 2.1. Section 2.2 provides a basic understanding of location-based services and outlines the most commonly used LBSs. Section 2.3 surveys the related work and categorizes them into location privacy and private proximity detection.

2.1 Privacy

Privacy is a concept that has been studied in various fields for more than 100 years. In [pri02], many definitions and aspects of privacy are surveyed. Since the 1960s, information technology has had a significant impact on the way that we live, therefore, information privacy has been extensively investigated during the last decades. Information privacy bridges the gap between massive data collection/processing and the legal/political/technological issues surrounding them.

It is obvious that technological advancements provide our society with numerous advantages. However, employing them in everyday life raise serious concerns about misusing of users' data. Nowadays, huge amounts of data is created about people while they browse the WWW, connect with their friends using online social networks or employ their mobile phones. Mobile devices are typically equipped with position sensors (e.g., GPS receivers) to process physical location. Processing such data is very critical as it might disclose sensitive information about the private lives of users.

Since the beginning of the 21st century, tremendous interest in user data has resulted in the existence of many companies whose core business are formed by collecting data, processing it and monetizing the result. It is estimated that about 50 billion devices will be connected to the Internet in 2023. Consequently, electronic devices are able to communicate with each other resulting in the creation of more data. Therefore, artificial intelligence algorithms have enough data to analyze every single part of our lives. Users are typically unaware to figure out how and by whom their data is being processed [Her16].

It should be noted that in our society, the necessity for exchanging information can be seen as a building block for connecting people together, getting access to useful services and many more. For example, people need to reveal their address and bank account information for gym registration, tell their friends about their holiday plans and so on. A user wants to share her information only with intended receivers.

According to Westin [Wes67], privacy is related to informational self-determination. Individuals should have the right to decide which information about them in which situations should be divulged. Therefore, information privacy is a relational concept that depends on the entities being involved.

2.1.1 Privacy Protection

According to Danezis and Gürses [DG10], all proposed *Privacy Technologies* can be separated into *Privacy Enhancing Technologies* (PETs) and *Privacy-Preserving Data Publishing and Mining*. The latter includes technologies that analyze personal information databases in a privacy-preserving manner. This is mostly achieved by employing cryptographic primitives. While PETs investigate technologies such as *anonymizers* which make users anonymous within a set of others. Tor [DMS04] is the most widely used anonymizer.

Authors in [DG10] categorize PETs into three categories as follows: privacy as confidentiality, privacy as control and privacy as practice. Privacy as confidentiality provides users with different technologies to prevent information from being revealed to the public. For example, the *Advanced Encryption Standard* (AES) can be utilized to provide confidentiality of the message content. While Tor provides confidentiality of the message content, it also protects its users from being monitored. Privacy as control refers to informational self-determination. *Identity Management Systems* (IMS) and *Single Sign-On* (SSO) systems are the most widely used technologies that provide users with the possibility to control which information of them should be revealed to which entity. Finally, privacy as practice can be seen as an extension to privacy as control. Users not only can decide about what information is revealed to which service, but they also are capable of understanding how information is being transmitted, aggregated and used.

In [BN11], Brunton and Nissenbaum propose *obfuscation* as another technique related to privacy as confidentiality, in order to protect user privacy. In obfuscation, users hide their data among a set of false information in order to make data collecting less valuable and less reliable. Therefore, it is hard to learn about users' sensitive information.

2.2 Location-Based Services

Location-based services (LBSs) are services that use the location of the client for adding value to the service. The added value could be filtering of information and selecting the neighboring point of interests, or activating the service when a user enters or leaves a predefined location [Küp05].

Location-based services can be categorized into *reactive* and *proactive* LBSs [Küp05]. In reactive LBS, the user invokes the service and establishes a service session. Then he requests for certain information, for example, a list of nearby gas stations, whereupon the service collects and processes location data and sends the result back to the user. Thus, a reactive LBS is activated by the user and characterized by a synchronous interaction between user and service. Proactive LBSs, on the other hand, are activated automatically when the user enters or leaves a certain point of interest. For example, an electronic tourist guide which informs people via SMS as soon as they approach a landmark. In proactive LBSs, users are continuously tracked by the service in order to inform them about some location events.

Furthermore, there are two types of queries that can be issued to a location-based server: *snapshot* and *continuous* queries. Examples of snapshot queries include *"find the closest restaurant"* or *"where is nearest post office within one mile of my location"*. In continuous queries, users continuously provide the server with their accurate location in order to obtain the precise answer for their queries, for example, *"report the location of the closest parking lot while driving a car"*.

In LBSs, the lack of sufficient controls may disclose the individuals' personal information which causes privacy concerns.

Design of Private Location-Based Services

Protecting users' privacy is the most challenging part in designing a private LBS, therefore, most private LBSs are designed for a specific use-case and cannot be reused for other use-cases. Moreover, users cannot trust in all cases the service provider and do not share their accurate location with it. In most private LBSs, the service provider is assumed to be *honest-but-curious*, thus it tries to learn as much as possible about its users. In the following, the more commonly used private LBSs are outlined [Her16].

- **Geo-Social Networks (GSNs)** employ both functionalities of LBSs and online social networks to provide their users with the ability to share their

whereabouts with friends. Other features of GNSs are to find nearby contacts and providing clients with some recommendations related to points of interest in their vicinity. However, the core functionality of any GSNs is location-sharing. In such services, for example, Alice and Bob wish to share their location with each other without revealing the exact location to a malicious party. In [Her16], the author surveys a variety of proposed techniques aimed at location privacy protection while users share their exact location with each other.

- **Friend-Nearby Notification Services** are designed in such a way to inform users about whether a friend of them is in close proximity [ZGH07]. In these services, for example, Alice and Bob engage in a private protocol in order to learn if they are close to each other or, in some proposed approaches, they even learn the accurate location of the other party. A friend-nearby notification service should detect and prevent users' misbehavior. For example, if Alice lies about her location thus she can learn about Bob's location without disclosing her own location. Consequences of such misbehavior lead to privacy violations.

- **Traffic Monitoring Services** are another type of common LBSs which offer a statistical infrastructure in order to route traffic in a more efficient way [HGXA06]. Each vehicle is equipped with some sensors and tracking devices to report to a LBS some information, such as its current speed, continuously. The LBS analyzes the information received from vehicles to compute statistics that can be used to navigate cars as well as detecting traffic jams, etc. However, since these applications require frequent updates of the user's position, the user's privacy may be violated by a *location tracking attack*. If an attacker tracks a vehicle over the time, he can find some confidential information about its driver.

- **Point-of-Interest (POI) Finder** is one of the most popular LBSs that allows mobile users to look for interesting places, such as restaurants or shopping centers. In POI finder, the user issues a query like *'Where is the nearest restaurant'* or *'what are the gas stations within one mile of my location'*. In response, a list of candidates is forwarded to the user.

2.3 Related Works

In this section, related work on location privacy and private proximity detection are reviewed. Most related and recent techniques designed to provide location privacy for LBS users can be categorized into two main groups: (i) schemes which rely on Trusted Third Parties (TTP-based schemes) and (ii) TTP-free schemes.

The main idea of TTP-based schemes is to utilize a trusted third party as a location *anonymizer* in order to blur users' location information before sending them to LBSs. Most proposed architectures in this category employ *k-anonymity* model [Swe02, GL08] and spatial cloaking techniques [CML06, CML11, WW10].

In *k-anonymity* model if the probability of recognizing the user who queries an LBS does not exceed $1/k$, then a query is considered private. The main responsibility of an anonymizer is to construct a *Cloaking Region* (*CR*) including a query user and $k-1$ other subscribed users in her vicinity. Therefore, the locations of k users will be masked into a cloaked area, and are indistinguishable. Spatial cloaking techniques are generally used for generating *CR* by an anonymizer to protect the privacy of LBS users. However, this approach has some weaknesses. If k users are in the same location, such as an airport, their location might be disclosed to an attacker. Besides, the anonymizer is considered as a single point of attack. Therefore, if an attacker gains access to it, users' location privacy is violated.

In TTP-free schemes, two main components at query time are users and the LBS provider. In [GKK$^+$08] Ghinita et al. propose two protocols for implementing a POI finder in a privacy-preserving way without using a trusted party. Their method is based on the *Private Information Retrieval (PIR)* [CKGS98, KO97] theory that makes it possible for a client to privately query a database on the server for a particular piece of information without revealing the query to the server. Both protocols employ space-partitioning data structures like *kd-tree* or *R-tree* to transform LBS queries to index-based queries. However, the drawback of their solution is that PIR is computationally heavy on the service provider.

Most related works on private proximity detection, such as *FriendLocator* [ŠTŠ$^+$09] and *VicinityLocator* [ŠTŠY10], partition the whole space into a fixed number of cells. Therefore, the proximity detection problem is turned into the *equality testing* problem aiming to determine whether two parties are located inside the same or nearby cells. Zhong et al. [ZGH07] proposed three protocols, namely *Louis*, *Lester*, and *Pierre* in order to compute the distance between two parties using the homomorphic

encryption. *Longitude* [MBF09] is another proposed privacy-preserving protocol for proximity detection that is based on a centralized architecture.

More similar to our work are protocols for *secure point inclusion problem* which was first presented by Atallah and Du [AD01]. Their suggested protocol for solving the point inclusion problem is based on two sub-protocols: *secure two-party scalar product* and *secure two-party vector dominance* protocol. In the former, Alice has a vector $A = (a_1, ..., a_n)$ and Bob has a vector $B = (b_1, ..., b_n)$. After the protocol execution, Alice (not Bob) gets $A \cdot B + v$ where v is a random secret known only to Bob. Their proposed solution is as follows. First, vector A is divided into m random vectors $V_1, ..., V_m$ such that $A = \sum_{i=1}^{m} V_i$. For each V_i, Alice sends p vectors to Bob, only one of which is V_i and the rest are arbitrary. Accordingly, Bob computes the scalar products between each of these p vectors and B. Alice then employs 1-out-of-p oblivious transfer protocol [Gol98] to obtain $V_i \cdot B + r_i$ from Bob, where r_i is a random number and $v = \sum_{i=1}^{m} r_i$. Finally, Alice can calculate $\sum_{i=1}^{m}(V_i \cdot B + r_i) = A \cdot B + v$. It should be pointed out that the idea behind this approach is utilized in our work presented later (Section 4.3.2). Similarly, in the secure two-party vector dominance protocol, Alice wants to privately check whether vector A dominates vector B. If A dominates B, then for all $i = 1, ..., n$, we have $a_i > b_i$. Their solution for this problem is based on Yao's Millionaire protocol [Yao82].

By utilizing the two aforementioned protocols, Atallah and Du [AD01] proposed one $O(n)$ protocol for the point-location problem where n is the number of edges of the polygon. Their algorithm in a straightforward way without worrying about privacy concern is as follows. Assume Alice has a point $A = (\alpha, \beta)$ and Bob has a polygon with n edges. The algorithm starts by finding the leftmost vertex l and rightmost vertex r of the polygon. Then all edges of the polygon are divided into two boundaries by a diagonal between l and r. Therefore, if the point A is above all the edges of the lower boundary and below all the edges of the upper boundary, then the point A is located inside the polygon, otherwise it is outside. It should be noted that this solution only covers the convex polygonal domains. Besides, since the vector dominance protocol involves several executions of Yao's Millionaire protocol, their approach is inefficient and computationally expensive. However, their protocol has been recently employed in [SS05] to privately determine whether a user is located inside the sensing area of a pervasive sensor network.

In [Tho09], Thomas studied the point inclusion problem in a star-shaped domain and a more general polygonal domain. He proposed three protocols including two

protocols for the star-shaped domain with round complexities $O(n)$ and $O(\log n)$ respectively, and one protocol for the more general polygonal domain with round complexity $O(n)$, where n is the number of vertices of the given polygon. His algorithm for the star-shaped domain is as follows. Assume P is a polygon with vertices P_i, for $1 \leq i \leq n$, which are named in the counter-clockwise direction. A star-shaped domain P always contains a point Q such that all line segments joining Q to P_i, for $1 \leq i \leq n$, are completely inside P. The next step employs binary search algorithm in order to find the wedge bounded by the rays $\overrightarrow{QP_i}$ and $\overrightarrow{QP_{i+1}}$ in which a given point M lies. After finding P_i and P_{i+1}, then M is inside P if the angle between P_i, P_{i+1} and M is a left turn. More details about his work can be found in [Tho09].

Mu et al. [MB16] solved the point-inclusion problem for arbitrary convex polygonal domains. Their approach employs a secure two-party computational protocol which is based on Paillier homomorphic cryptosystem in order to test whether a given point lies inside a convex polygon or not. Besides, they suggested a solution for handling an arbitrary concave polygon by partitioning the concave polygon into the minimum number of convex pieces using computational geometry algorithms. Therefore, their scheme can be invoked multiple times to evaluate the proximity query. In contrast to their protocol, our scheme can be employed to all complex polygonal domains (i.e, all arbitrary concave or convex polygons) without any additional cost.

In [LD05] Li et al. studied the point-location problem for a *circular domain*. However, their solution reveals some additional information about each party's location to the other party. Moreover, their solution is highly inefficient. Later, an efficient protocol for point-circle inclusion problem was proposed by Luo, Huang and Zhong [LHZ07] which is based on a two-round protocol for calculating the distance between two private points. In their protocol, only one party gets the output and the other party should trust the knowledgeable one for the output. Moreover, in [JKS+18] multiple privacy-preserving location proximity schemes are proposed which are applicable to the point-circle inclusion problem. Other similar works are discussed in [YSZ11, YSZ12, YLWY10].

In this thesis, we consider the point-inclusion problem in complex polygonal domains and propose two privacy-preserving schemes. Our solution provides a significant reduction in computational and communication complexity compared to the similar work.

3 Preliminaries

This chapter introduces the cryptographic and geometric primitives as well as basic concepts in Linear Algebra which will be utilized throughout the rest of this thesis.

3.1 Attacker Model

In the most general setting, we consider n parties jointly running a cryptographic protocol. We will assume that a potential attacker has corrupted a subset of the participating parties, has access to their data and controls the way they interact with the other parties when running the protocol. We will distinguish between two types of attackers (adversarial behavior) which will be described in this section.

3.1.1 Semi-honest Attacker Model

A *semi-honest* (known also as *honest-but-curious*) adversary follows the protocol correctly and does not send any fake information, however, he attempts to deduce as much information as possible from exchanged messages during the protocol execution. For example, he might record all intermediate computations to learn something from the other parties, but he does not act to corrupt, replace or drop messages. This adversary model is discussed in detail in [Gol09, Chapter 7.2].

3.1.2 Malicious Attacker Model

Another type of attacker is *malicious* adversary (known also as an *active* attacker). A malicious attacker attempts to manipulate the normal execution of the protocol in order to infer some susceptible information about the other parties. Moreover, he might try to send corrupted information to prevent the protocol from terminating with an accurate output. A well known and widely used way for transforming a secure protocol in the semi-honest adversary model into one that is secure against malicious attackers is presented in [Gol09, Chapter 7.4]. This is accomplished by forcing the active attacker to behave in a semi-honest manner. Typically, protocols secure against active adversaries are not as efficient as protocols that are secure in the semi-honest attacker model. More details can be found in [Gol09, Chapter 7.4].

3.2 Cryptographic Preliminaries

The cryptographic concepts utilized as a building block into our work are defined in the following. Readers who are interested in more details can use *Handbook of Applied Cryptography* [KMVOV96] as a reference.

3.2.1 1-out-of-N Oblivious Transfer

In this thesis, we design one protocol for the point-inclusion problem that considerably depends on 1-out-of-N Oblivious Transfer protocol [BCR86, EGL85]. At the beginning of this protocol, one party (Alice) has N inputs $A_1, ..., A_N$ and the other party (Bob) is interested to learn the input A_i, where $1 \leq i \leq N$. At the end of the protocol, Bob only learns the input A_i without learning anything about the other inputs and without allowing Alice to know which input has been retrieved. In [NP99] Naor and Pinkas propose an efficient 1-out-of-N Oblivious Transfer protocol. Recently, the simplest and most efficient protocol for 1-out-of-N OT is presented in [CO15] which is a simple tweak of the Diffie-Hellman key-exchange protocol.

3.2.2 Homomorphic Encryption

Homomorphic encryption demonstrates a group of semantically secure encryption functions that allow performing certain algebraic operations directly on the ciphertext without first decrypting them. Many public-key cryptosystems are partially homomorphic. It means that they facilitate the evaluation of addition or multiplication directly on the ciphertext space. In this thesis, we employ *additively* homomorphic encryption which has the following properties. First, given two encryptions $E(m_1)$ and $E(m_2)$, we can compute the encryption of $m_1 + m_2$ by multiplying their encryptions.

$$E(m_1 + m_2) = E(m_1) \cdot E(m_2)$$

Second, an encrypted value $E(m)$ can be multiplied by a constant k as follows:

$$E(k \cdot m) = E(m)^k$$

In our implementation, we employ Paillier [Pai99] which is an additively homomorphic cryptosystem.

3.2.3 The Paillier-Encryption Scheme

The Paillier cryptosystem [Pai99] is an additive homomorphic public-key cryptosystem which was invented by Pascal Paillier in 1999. In this section, we first describe the key generation process, the encryption and the decryption operation. We then demonstrate the homomorphic properties of the scheme.

Key Generation

1. Compute $N = pq$, where p and q are randomly selected large prime numbers.
2. Compute $\lambda = lcm(p-1, q-1)$, where lcm means Least Common Multiple.
3. Select a generator g where $g \in \mathbb{Z}^*_{N^2}$ so that the order of g is a nonzero multiple of N. This can easily be validated by checking the equation

$$\gcd(L(g^\lambda \bmod N^2), N) = 1$$

   where function L is defined as $L(u) = \dfrac{u-1}{N}$.
4. The public key is the pair (N, g).
5. The private key is λ.

Encryption

1. We want to encrypt a plaintext $m \in \mathbb{Z}_N$.
2. Select a random number $r \in \mathbb{Z}_N$.
3. Compute the ciphertext c using $c = g^m r^N \mod N^2$.

It should be noted that while we choose a plaintext in $\mathbb{Z}_N$, we generate a ciphertext modulo N^2. Thus, the ciphertext space is $\mathbb{Z}^*_{N^2}$.

Decryption

1. We want to decrypt a ciphertext $c \in \mathbb{Z}^*_{N^2}$.
2. Compute the plaintext m using $m = \dfrac{L(c^\lambda \bmod N^2)}{L(g^\lambda \bmod N^2)} \mod N$.

Note that Paillier cryptosystem is probabilistic, i.e., ciphertexts of a plaintext m differ between encryptions with high probability. This is due to the use of different

random blinding factor r. As a result, Paillier scheme is resistant to *dictionary attacks*.

Homomorphic Properties Paillier scheme is additively homomorphic on $\mathbb{Z}_N$. This is easy to prove. Given two ciphertexts $c_1, c_2 \in \mathbb{Z}^*_{N^2}$, we know that there exist $m_1, m_2 \in \mathbb{Z}_N$ and $r_1, r_2 \in \mathbb{Z}_N$ such that $c_1 = g^{m_1} r_1^N \bmod N^2$ and $c_2 = g^{m_2} r_2^N \bmod N^2$. Now we have:

$$\begin{aligned} D(c_1 \cdot c_2) &= D(g^{m_1} r_1^N \bmod N^2 \cdot g^{m_2} r_2^N \bmod N^2) \\ &= D(g^{(m_1+m_2)} (r_1 r_2)^N \bmod N^2) \\ &= m_1 + m_2 \\ &= D(c_1) + D(c_2) \end{aligned}$$

The reason why we utilize the Paillier encryption scheme in our privacy-preserving schemes is the following equations:

$$\forall m_1, m_2 \in \mathbb{Z}_N \quad \text{and} \quad k \in \mathbb{N}$$

$$D(E(m_1)E(m_2) \bmod N^2) = m_1 + m_2 \bmod N$$

$$D(E(m)^k \bmod N^2) = km \bmod N.$$

Another secure and efficient additively homomorphic encryption scheme that could be employed instead is DGK [DGK07].

3.2.4 Encrypting Negative Integers

The message space $\mathbb{Z}_N$ consists of positive integers only, hence in order to represent negative integers, we take advantage of the cyclic property of the Paillier cryptosystem. If the message space is $m \in \mathbb{Z}_N$, then $N - m \equiv -m \mod N$. Therefore, we use the top half of the message space to denote negative numbers. As a result, $-m$ is represented by $N - m$. For this purpose, the message space is interpreted as the complement arithmetic representation for N-bit integers. It should be pointed out that in the case of an overflow, a positive number suddenly becomes a negative number or vice-versa. In order to avoid overflows, we should ensure that two intervals are disjoint (see Figure 1).

Assume that $\alpha \in \mathbb{Z}_{N'}$, where $N' < N$. Let $S_1 = \{0, 1, ..., N'\}$ and $S_2 = \{N - N', ..., N - 1\}$ denote positive and negative numbers respectively. An adequate condition to ensure that $S_1 \cap S_2 = \emptyset$ is:

$$N' < N - N' \Leftrightarrow N' < \frac{N}{2}$$

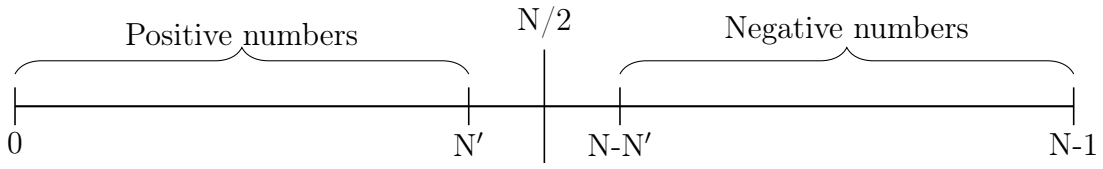


Figure 1: Using the cyclic property of the cryptosystem to represent negative numbers [GKKB09].

3.2.5 Additive Blinding

Additively blinding the value x means producing a uniformly random element $y \in \mathbb{Z}_N$ such that any information about the value x is not disclosed by y. At first, a random element $r \in \mathbb{Z}_N$ is chosen, where the distribution of r is the discrete uniform distribution. In this case, the probability of choosing a random variable R is $P(R = r) = \frac{1}{N}$ for all $r \in \mathbb{Z}_N$. Then we calculate $y = r + x \mod N$. Since the element y is a uniformly random element of the set $\mathbb{Z}_N$, no information about the value x is revealed by y.

3.3 Geometric Preliminaries

In the following, we give necessary preliminary geometric definitions and introduce basic concepts in linear algebra. For more details on these definitions see [LL09, PS88, DBVKOS97, Cor08].

3.3.1 Vectors in Euclidean Space

In the Euclidean coordinate system, a vector $\overrightarrow{PQ}$ is a directed line from a point P (initial point) to a point Q (terminal point), in which the length of the directed line is defined as the vector's *magnitude*, denoted by $|\overrightarrow{PQ}|$, and its *direction* is the same as the direction of the drawn line. In addition, each point $P = (x, y)$ also can represent one vector in the Cartesian system, where the tail of the vector is at the origin of the Cartesian system and the head of the vector is at the point P. A zero vector is a vector such that its initial and terminal points are the same. In fact, a zero vector is just a point in the Cartesian system. Unless otherwise indicated, whenever we refer to a vector as $\vec{v} = (a, b)$, we mean a vector starting at the origin of the Cartesian system and ending at the point (a, b).

Two nonzero vectors are *equal* if the magnitude and direction of them are the same. Therefore, two vectors with different initial points but with the same magnitude and direction are considered equal. By this definition, the vector $\overrightarrow{PQ}$ is equal to the vector $\vec{v}$ whose initial point is at $(0, 0)$ and its terminal point is $(Q_x - P_x, Q_y - P_y)$.

3.3.2 Geometric Definition of Dot Product

The dot product of two vectors is one type of multiplication of two vectors which results in a scalar. The dot product of $\vec{v} = (v_1, v_2)$ and $\vec{w} = (w_1, w_2)$, denoted by $\vec{v} \cdot \vec{w}$, is defined as below:

$$\vec{v} \cdot \vec{w} = v_1 w_1 + v_2 w_2$$

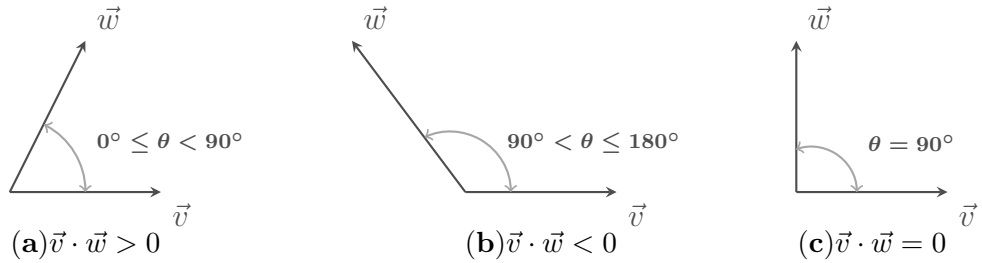
It is possible to build a relationship between the dot product and the angle between two vectors with the same initial point. Therefore, the geometric definition of the dot product can be determined as:

$$\vec{v} \cdot \vec{w} = |\vec{v}|\,|\vec{w}| \cos\theta$$

where θ is the smallest nonnegative angle between two nonzero vectors, so that $0° \leq \theta \leq 180°$. Since $\cos\theta > 0$ for $0° \leq \theta < 90°$ and $\cos\theta < 0$ for $90° < \theta \leq 180°$, we have:

$$\vec{v} \cdot \vec{w} \text{ is } \begin{cases} > 0 & \text{for } 0° \leq \theta < 90° \\ 0 & \text{for } \theta = 90° \\ < 0 & \text{for } 90° < \theta \leq 180° \end{cases}$$

As depicted in Figure 2, the dot product of two vectors $\vec{v}$ and $\vec{w}$ is positive, negative, or zero, depending on whether the non-negative angle between them is acute, obtuse, or a right angle, respectively. However, the exact size of the angle cannot be deduced by the actual numerical value of the dot product.



(**a**)$\vec{v} \cdot \vec{w} > 0$ (**b**)$\vec{v} \cdot \vec{w} < 0$ (**c**)$\vec{v} \cdot \vec{w} = 0$

Figure 2: Sign of the dot product and angle between two vectors [Cor08].

3.3.3 Geometric Definition of Determinant

A 2×2 matrix is an array of two rows and two columns, denoted as:

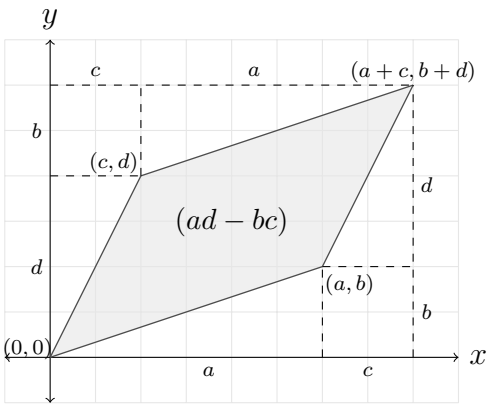
$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \text{ or } \begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

where a, b, c, d are scalars. The determinant of a matrix A, denoted by $det(A)$ or $|A|$, is a scalar value that can be calculated from the elements of such a matrix and is defined by the following formula:

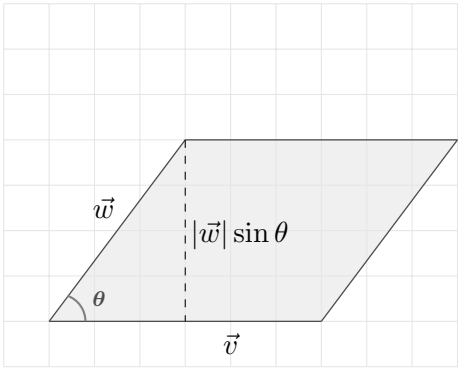
$$|A| = \begin{vmatrix} a & b \\ c & d \end{vmatrix} = ad - bc.$$

Each row of a matrix can represent one vector. For the 2×2 matrix above, the vectors are $\vec{v} = (a, b)$, and $\vec{w} = (c, d)$. Assume P be a parallelogram with adjacent vectors $\vec{v} = (a, b)$ and $\vec{w} = (c, d)$, then the *signed area* of the parallelogram with vertices at $(0, 0)$, (a, b), (c, d), and $(a + c, b + d)$, shown in Figure 3a, is computed as below:

$$\text{area}(P) = (a+c)(b+d) - 2(\frac{1}{2}ab) - 2(\frac{1}{2}cd) - 2bc = ab + ad + cb + cd - ab - cd - 2bc = ad - bc.$$



(a) The signed area of the parallelogram is equal to the determinant of the matrix built of two vectors forming two sides of the parallelogram.



(b) The parallelogram with adjacent vectors $\vec{v} = (a, b)$, $\vec{w} = (c, d)$ and angle θ between them has area $|\vec{v}||\vec{w}| \sin \theta$.

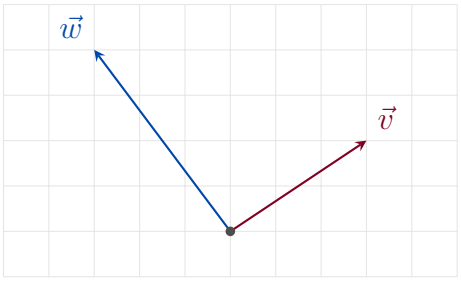
Figure 3: Signed area $= |\vec{v}||\vec{w}| \sin \theta = ad - bc$.

Swapping the rows changes the sign of the determinant $\begin{vmatrix} a & b \\ c & d \end{vmatrix}$. This is why the area is determined as "signed". Geometric interpretation of the negative area is that the

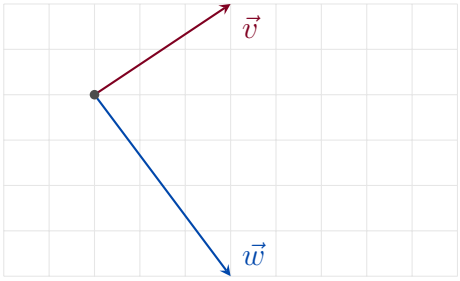
rotation angle from the first vector to the second vector is in a clockwise direction. As a result, the sign of the area changes when the vertices are listed in a different order. Note here that the rotation angle θ between first vector $\vec{v}$ to second vector $\vec{w}$ is always chosen in such way that $-180° < \theta \leq 180°$.

As shown in Figure 3b, the signed area of the parallelogram can also be calculated using the sine formula where the angle of rotation (θ) from $\vec{v}$ to $\vec{w}$ ranges from $-180°$ to $180°$, with negative angles corresponding to clockwise rotation, and positive angles corresponding to counterclockwise rotation.

As illustrated in Figure 4, the sine formula can help us to predict the sign of the determinant since $\sin\theta$ is positive for $0° < \theta < 180°$ and negative for $-180° < \theta < 0°$. If two vectors point in either the same direction or exact opposite directions then the corresponding determinant is zero.



(a) The determinant $\begin{vmatrix} v_x & v_y \\ w_x & w_y \end{vmatrix}$ is positive if $\vec{w}$ is in the counterclockwise rotation angle from $\vec{v}$.



(b) The determinant $\begin{vmatrix} v_x & v_y \\ w_x & w_y \end{vmatrix}$ is negative if $\vec{w}$ is in the clockwise rotation angle from $\vec{v}$.

Figure 4: Sign of determinant of two vectors $\vec{v}$ and $\vec{w}$.

3.3.4 Smallest-Signed Angle Between Two Vectors

Assume that we have two vectors $\vec{v} = (v_x, v_y)$ and $\vec{w} = (w_x, w_y)$. In our proposed privacy-preserving schemes for the point inclusion problem, we are interested to compute a signed magnitude of the smallest angle that takes $\vec{v}$ to $\vec{w}$ or vice-versa. The common mistake for calculating the angle between two vectors is shown in Figure 5 (the right one). As shown in Figure 5, the signed magnitude of the angle that takes $\vec{v}$ to $\vec{w}$ is -90°, while the angle for taking $\vec{w}$ to $\vec{v}$ is $+90°$.

As discussed earlier, the dot product and the determinant of two vectors $\vec{v}$ and $\vec{w}$

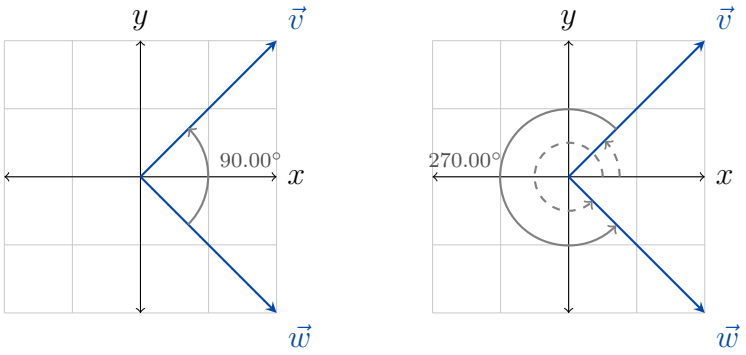


Figure 5: Angle between two vectors

are defined as follows.

$$\vec{v} \cdot \vec{w} = v_x w_x + v_y w_y = |\vec{v}|\,|\vec{w}| \cos(\theta) \tag{1}$$

$$\det(\vec{v}, \vec{w}) = v_x w_y - v_y w_x = |\vec{v}|\,|\vec{w}| \sin(\theta) \tag{2}$$

From Equations 1 and 2, the smallest-signed angle that takes $\vec{v}$ to $\vec{w}$ is calculated as follows.

$$\theta = \arctan(\frac{\det(\vec{v}, \vec{w})}{\vec{v} \cdot \vec{w}}) \tag{3}$$

where the *arctan* function can be computed using *atan*2 function in many programming languages. The *atan*2 is a two-argument function that is often written as $\theta = \tan^{-1}(\frac{y}{x})$. The $atan2(y, x)$ calculates the angle in radian between the positive x-axis of the Cartesian system and the point given by the coordinate (x, y) [USD11]. However, *atan*2 function has discontinuity when the output angle crosses π. The angle is positive for counter-clockwise angles ($y > 0$), and negative for clockwise angles ($y < 0$). Thus, it produces the result in the range $-\pi < \theta \leq \pi$ (Figure 6).

The definition of the *atan*2 is as follows [USD11].

$$atan2(y, x) = \begin{cases} \arctan\left(\frac{y}{x}\right) & \text{if } x > 0 \\ \pi + \arctan\left(\frac{y}{x}\right) & \text{if } y \geq 0, x < 0 \\ -\pi + \arctan\left(\frac{y}{x}\right) & \text{if } y < 0, x < 0 \\ \frac{\pi}{2} & \text{if } y > 0, x = 0 \\ -\frac{\pi}{2} & \text{if } y < 0, x = 0 \\ \text{undefined} & \text{if } y = 0, x = 0 \end{cases}$$

For instance, the angle from the x-axis to the vector $(2, 2)$ is $\pi/4$ rad or $45°$, while the angle from the x-axis to the vector $(-2, -2)$ is $-3\pi/4$ rad or $-135°$. Therefore, signs of two vectors are taken into account while using the *atan*2 function.

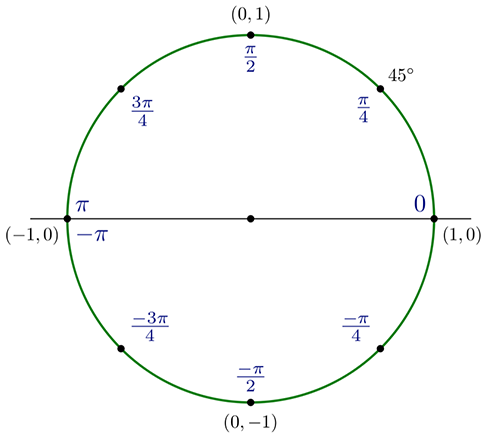


Figure 6: The four-quadrant inverse tangent, $atan2(y, x)$.

3.3.5 Blinding an Angle Between Two Vectors

The inverse tangent function satisfies the addition formula as follows.

$$\arctan(\alpha) + \arctan(\beta) = \arctan\left(\frac{\alpha + \beta}{1 - \alpha\beta}\right) \tag{4}$$

As discussed in Section 3.3.4, the smallest-signed angle that takes $\vec{v}$ to $\vec{w}$ is $\theta = \arctan(\frac{\det(\vec{v},\vec{w})}{\vec{v}\cdot\vec{w}})$. Therefore, in order to add randomness (i.e., θ') to the exact signed angle θ, we can choose two random numbers r_x and r_y, such that $\theta' = \arctan(\frac{r_y}{r_x})$. According to Equation 4, the randomized angle $\theta + \theta'$ is calculated as follows.

$$\underbrace{\arctan\left(\frac{\det(\vec{v},\vec{w})}{\vec{v}\cdot\vec{w}}\right)}_{\theta} + \underbrace{\arctan\left(\frac{r_y}{r_x}\right)}_{\theta'} = \underbrace{\arctan\left[\ \frac{(\det(\vec{v},\vec{w}))\cdot r_x + (\vec{v}\cdot\vec{w})\cdot r_y}{(\vec{v}\cdot\vec{w})\cdot r_x - (\det(\vec{v},\vec{w}))\cdot r_y}\ \right]}_{\theta+\theta'} \tag{5}$$

As we know, $atan2$ function has a discontinuity when the output angle crosses π, which can lead to discontinuity in $\theta + \theta'$. It means that sometimes the sum of two positive angles could be negative. Similarly sometimes the sum of two negative angles could be positive. For example, assume that $atan2(53, -105) \approx 153.21709°$and $atan2(60, 23) \approx 69.02650°$. According to Equation 5, we have:

$$atan2(53, -105) + atan2(60, 23) = atan2(-5081, -5595) \approx -137.75640°$$

In order to avoid that, we need to make sure that the added random angle θ' and the actual angle θ have different signs. As depicted in Figure 6, $atan2(y, x)$ and y have the same signs. Therefore, if $\det(\vec{v}, \vec{w}) \geq 0$, then r_y should be negative. Similarly, if $\det(\vec{v}, \vec{w}) < 0$, then r_y should be positive. The random number r_x can be either positive or negative.

3.3.6 Computational Geometry

Our proposed protocols borrow the angle summation algorithm from computational geometry. As such, some basic terms and concepts utilized throughout this thesis are defined in the following.

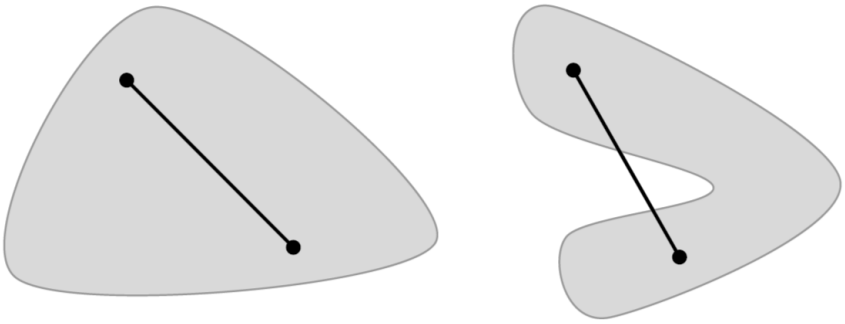


Figure 7: A Convex Region (left), and a Concave Region (right).

Convex and Concave Regions A convex region [DBVKOS97] has the property that each line segment between two distinct points of the region lies completely within the region, while in a concave region there is at least one line segment between two different points of the region which does not lie completely within the region. We show an example in Figure 7, illustrating convex region and concave region.

Jordan curve theorem In the *Jordan curve theorem* [BJMR75] a non-self-intersecting loop in the plane, which is called a *Jordan curve*, divides the plane into two regions: interior and exterior region. The former refers to the region enclosed by the curve and the latter contains all of the exterior points.

Point in polygon problem One interesting problem in the field of computational geometry is to check whether a given point M is enclosed by an arbitrary polygon P or not (this is a special case of point inclusion problem). In this thesis, we are only interested in non-self-intersecting polygons. Two proposed solutions for solving this problem are the *even-odd rule* and the *winding number* [HA01]. The former employs *ray-casting* and latter *angle summation* algorithm [Wei94]. In the following, both algorithms are described.

The ray-casting algorithm counts the number of times a ray, starting from the point M and going in any fixed direction, crosses the edges of P. If this ray crosses the edges of the polygon an odd number of times, the point is inside the polygon,

otherwise it is outside (see Figure 8).

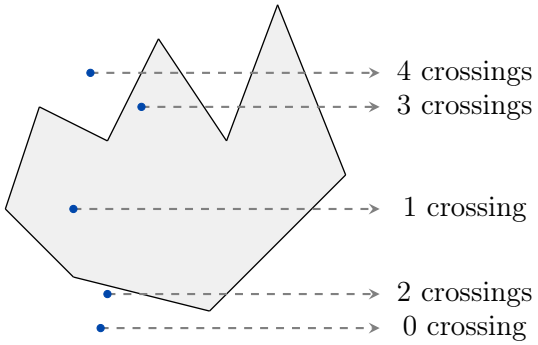


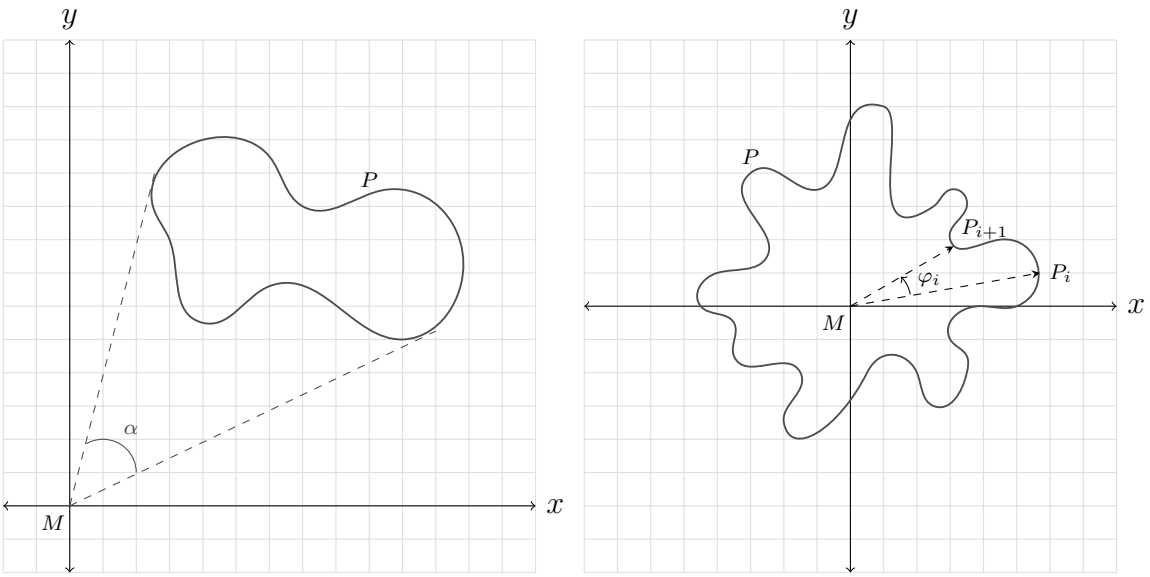
Figure 8: Ray-casting algorithm

The winding number algorithm counts the number of times that P travels around the point M. It can be shown that the point M lies inside the polygon, if the winding number is nonzero, otherwise it is outside. The winding number employs the angle summation algorithm in order to calculate the sum of all signed angles between the vertices of the polygon P, as viewed from the point M. Without loss of generality, we assume $M = (0, 0)$. As depicted in Figure 9b, the winding number of the point M with respect to the polygon P is $\omega(M, P) = \frac{1}{2\pi} \sum_{i=1}^{n} \varphi_i$, where φ_i is the signed angle between the edges $\overline{MP_i}$ and $\overline{MP_{i+1}}$.

In this thesis, we utilize the angle summation algorithm [Wei94] to derive the answer for the point in polygon problem. As mentioned already, we assume that P is a non-self-intersecting polygon. Therefore, as shown in Figure 9, if point M lies inside P, the angle summation algorithm results in plus or minus 360 degrees, depending on the chosen direction for naming the vertices of P. If the sum of angles is 0 degrees, then the point M is outside of P. This algorithm will be discussed in detail in Section 4.2.

3.4 Discretization

In practice, the latitude and longitude of any GPS location are floating point numbers. Latitude values (Y-values) range between -90 and $+90$ degrees, while longitude values (X-values) range between -180 and $+180$ degrees. On the other hand, Paillier encryption scheme requires the use of integers alone. Therefore, floating point numbers should be converted to integer values in order to use them as inputs in our privacy-preserving protocols. We map each floating point number to integer



(a) $\omega(M,P) = \frac{1}{2\pi}\sum_{i=1}^{n}\varphi_i = \frac{1}{2\pi}(\alpha-\alpha) = 0$

(b) $\omega(M,P) = \frac{1}{2\pi}\sum_{i=1}^{n}\varphi_i$

Figure 9: Winding number algorithm for the point in polygon problem.

according to the following formula: $\mathbf{discretize}_e(x) = \lfloor 10^e \cdot x + 0.5 \rfloor$. The e parameter controls the shift of the decimal point. Therefore, by choosing the higher e we can ensure that the more digits after the decimal point are preserved [ŠG14]. By choosing $e = 8$, we can pinpoint a location to within 1 millimeter.

3.5 Notation

Throughout this work we use the following notations. Let us assume that $M = (a, b)$ is a given point, and P is a polygon with vertices P_i, for $i = 1, 2, ..., n$. For two vectors $\overrightarrow{MP_i}$ and $\overrightarrow{MP_{i'}}$, where $i' = (i+1) \mod n$, the notations det_i and dot_i denote, respectively, the determinant and dot product of two given consecutive vectors in the (x, y) plane. Also, by writing $\langle m \rangle$ for some $m \in \mathbb{Z}_N$, we refer to the encryption of message m using a homomorphic encryption scheme (e.g. Paillier encryption). Moreover, $\langle m \rangle_{pk_A}$ specifies that the encryption is performed under the public key pk_A. To denote that a value m is to be encrypted, we also use the notation $E(m)$. To denote decryption, we write

$$m \Leftarrow \langle m \rangle \text{ or } D(\langle m \rangle).$$

4 Secure Protocols for Point Inclusion Problem

This chapter is dedicated to the development of two privacy-preserving protocols for the point-inclusion problem. Firstly, we define the system model and our design goal. Next, we study the proposed algorithm for solving the point-inclusion problem in complex polygonal domains without worrying about privacy concern. Finally, we utilize the aforementioned algorithm and develop two privacy-preserving proximity detection protocols that are applicable to arbitrary polygonal domains.

4.1 System Model and Design Goal

In this section, we describe our system model and security requirements for the suggested schemes, and identify our design goal.

4.1.1 System Model

Our system model is similar to the proposed system model in [ZWL+18] and consists of three main parts: 1) proxy server (PS); 2) query user (QU) who is looking for his/her friends in one specific area; and 3) user's friends (UFs) who are participating in multi-party communication for proximity detection. The system model is shown in Figure 10.

- PS is a proxy server, which acts as an intermediate layer among users while they are engaging in secure multi-party computational geometry to find their friends in one specific area. To enrich users' experiences and protecting their privacy, the exact locations of users should not be exposed to the proxy server. In our system, the main responsibility of PS is forwarding the encrypted and blurred data among users.
- QU is a user who is willing to find out which friends of her are within one specific region. At first, she generates her own friends' list and selects one polygonal domain on the map. Then, she queries her friends in the target area.
- UF are friends of QU. In the process of secure multi-party computational geometry, QU sends encrypted query information to each UF. After receiving the blurred data by UF, each UF uses his/her own geographic location and does a hybrid calculation with the encrypted data in order to get the query

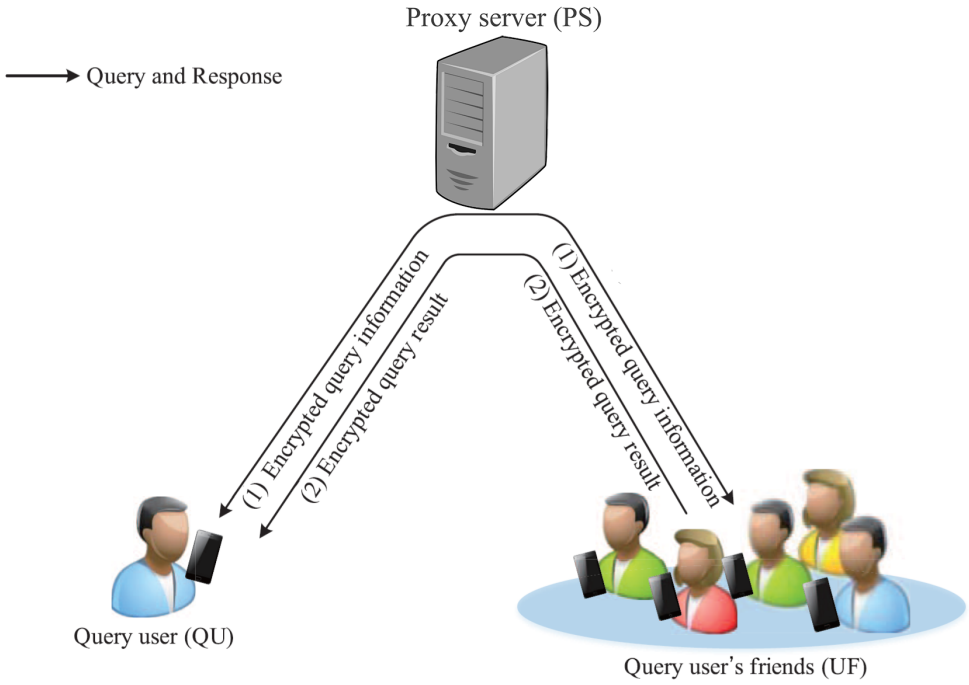


Figure 10: System model

result. Meanwhile, UF might communicate with QU to facilitate the hybrid calculation. Afterward, the encrypted query result is only decrypted by QU with further calculating.

4.1.2 Security Requirements

In our security model, the privacy of QU's query information and UF's exact location should be ensured. We assume that QU and UF are semi-honest (see Section 3.1). Therefore, QU and UF will follow the protocol correctly and will not send any fake information; however, they will try to gain access to each other's sensitive information through keeping a record of all intermediate computations and deriving the other party's input. In addition, different UFs might collaborate with each other in order to infer some sensitive information about QU's query. Moreover, PS is semi-honest, but it may try to obtain sensitive information of users from query requests and responses. Finally, we assume that the data will not be tampered and modified during the query process.

By considering the above security issues, in order to be successful in running a secure proximity detection protocol among users, the most important security requirement

that should be fulfilled is *privacy*. It means that during the query process, QU's geometric range query cannot be leaked to PS and UF. Meanwhile, UF's accurate location cannot be exposed to QU and PS. In addition, query results should be decrypted only by QU.

4.1.3 Design Goal

Considering the aforementioned system model and security requirements, we aim to design two efficient privacy-preserving proximity detection protocols with accurate results for arbitrary polygonal domains. Therefore, the following three objectives should be accomplished.

1. Privacy-preserving should be ensured: protecting users' location data against disclosure to other participants is the primary goal of our system design. Therefore, our proposed schemes should achieve confidentiality.

2. Accuracy of query results for arbitrary polygonal domains should be guaranteed: the users' experience is a significant aspect of our proposed schemes, and it is necessary that the accuracy of results should not be reduced while protecting users privacy.

3. Low communication overhead and computational complexity should be obtained: since smartphones have a limited battery, the improvement in communication and computation complexity can reduce energy consumption.

4.2 Algorithm

In this section, we first present the angle summation algorithm [Wei94] in plaintext domain, and then transform the operations into the encrypted domain. As mentioned in [Wei94], the angle summation algorithm is applicable to all complex polygonal domains without partitioning a complex polygon into the number of convex pieces. The algorithm takes as inputs a polygon P with vertices $P_i = (x_i, y_i)$, $1 \leq i \leq n$, where the vertices are named either in the clockwise or anticlockwise direction, and a point $M = (a, b)$. As defined in the following, the goal is to specify whether M lies inside P or not.

Definition 1. *A point M lies inside a polygon P with vertices P_i, if the sum of all signed angles that takes $\overrightarrow{MP_i}$ to $\overrightarrow{MP_{i'}}$, where $i = 1, 2, ..., n$ and $i' = (i + 1)$ mod n,*

is either $+360°$ *or* $-360°$ *where the chosen direction for listing the vertices of* P *is in the anticlockwise or clockwise order respectively. In the case that* M *is outside* P*, the sum of all signed angles mentioned earlier is* $0°$*.*

Initially, the algorithm starts by determining vectors from point M to all vertices of P. As mentioned in Section 3.3.1, a vector from point $M = (a, b)$ to $P_i = (x_i, y_i)$ denoted by $\overrightarrow{MP_i}$ where $1 \leq i \leq n$, is equal to the vector $\vec{v_i} = (x_i - a, y_i - b)$ whose initial point is at $(0, 0)$ and its terminal point is $P_i - M$. In this case, we can assume that the origin of the Cartesian system is shifted to the point $M = (a, b)$ without changing the orientation of the axes. As a result, the new coordinates of vertices under translations will be given by $X_i = x_i - a$ and $Y_i = y_i - b$.

In the next step, the algorithm calculates the smallest-signed angle between every two consecutive vectors according to the chosen direction for naming the vertices. Therefore, by computing the dot product and determinant of every two consecutive vectors, the signed angle between them is calculated using Equation 3.

Finally, the output is the sum of all signed angles computed in the previous step, and it depends on the chosen direction for labeling the vertices of P. As a result, the output is one of the following three cases:

$$\text{The output is } \begin{cases} +360° & \text{where } M \text{ is inside } P \text{ with anticlockwise-listed vertices;} \\ -360° & \text{where } M \text{ is inside } P \text{ with clockwise-listed vertices;} \\ 0° & \text{where } M \text{ is outside } P; \end{cases}$$

Remark. *In the case of* M *lying on the boundary of* P*, the algorithm may lead to ambiguities. In other word, the point* M *might be identified inside or outside of* P*, depending on the shape of* P *and the chosen direction for naming the vertices. In addition, if* M *coincides with one of the vertices of* P*, the algorithm mentioned above does not work correctly, because* $atan2(0, 0)$ *is undefined.*

All above steps are summarized here:

Algorithm 1 The angle summation algorithm for the point inclusion problem in complex polygonal domains.

Input: A polygon P with vertices $P_i = (x_i, y_i)$, $1 \leq i \leq n$, and a point $M = (a, b)$.

Output: Determining whether M lies inside P or not.

Algorithm steps:

1. Moving the origin of the Cartesian system from $(0, 0)$ to the point $M = (a, b)$. As a result, the new coordinates of vertices are $X_i = x_i - a$ and $Y_i = y_i - b$.

2. For each index i, the dot product and determinant of two consecutive vectors $\vec{v_i} = (x_i - a, y_i - b)$ and $\vec{v_{i'}} = (x_{i'} - a, y_{i'} - b)$, where $i = 1, 2, ..., n$ and $i' = (i + 1) \mod n$, are computed as below:

$$\vec{v_i} \cdot \vec{v_{i'}} = x_i x_{i'} - a(x_i + x_{i'}) + a^2 + y_i y_{i'} - b(y_i + y_{i'}) + b^2 \tag{6}$$

$$\det(\vec{v_i}, \vec{v_{i'}}) = x_i y_{i'} - x_{i'} y_i + b(x_{i'} - x_i) + a(y_i - y_{i'}) \tag{7}$$

For simplicity, we denote the dot product and the determinant of two vectors $\vec{v_i}$ and $\vec{v_{i'}}$ by dot_i and det_i respectively.

3. The final result is the sum of all smallest-signed angles between every two consecutive vectors $\vec{v_i}$ and $\vec{v_{i'}}$, and is calculated as follows.

$$\text{output} = \sum_{i=1}^{n} atan2\big(det_i, dot_i\big)$$

For better presentation, we examined the angle summation algorithm for different types of polygons shown below. The algorithm works accurately for complex polygons.

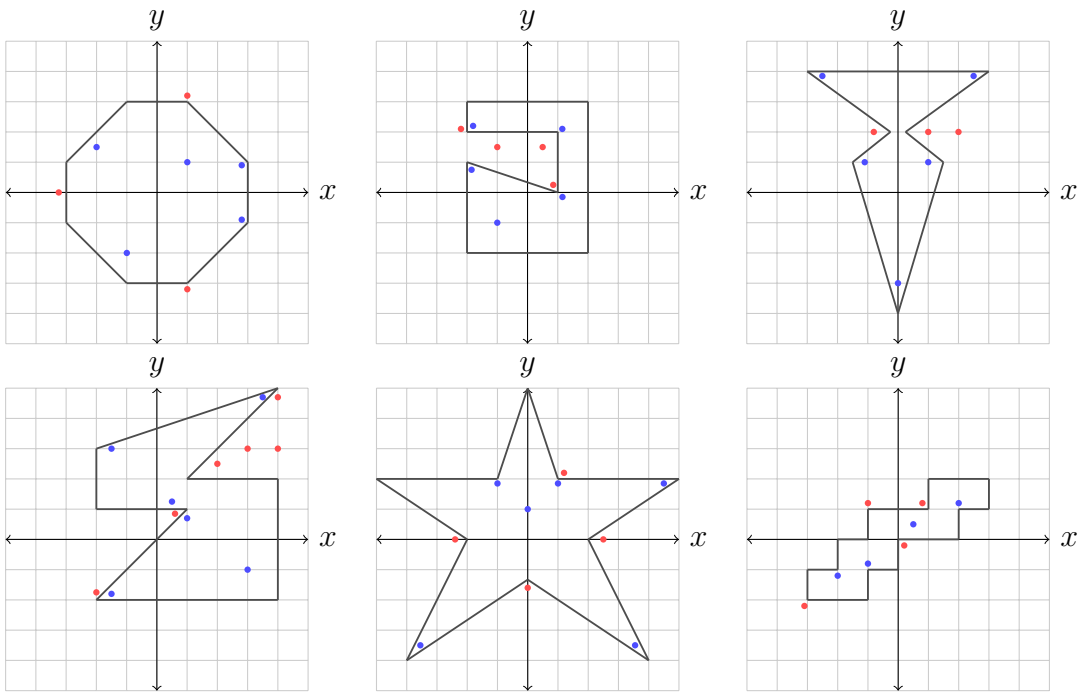


Figure 11: Testing the algorithm with different types of polygons, all blue points are recognized inside, and all red points are recognized outside the polygons.

4.3 The Proposed Privacy-Preserving Schemes

In this section, we utilize the algorithm described in Section 4.2, as a building block, in order to present two privacy-preserving solutions for the point-inclusion problem in complex polygonal domains. To the best of our knowledge, this is the first time that the proposed privacy-preserving solutions are applicable to all concave and convex polygons (except self-intersecting polygons). Moreover, our first solution provides a significant reduction in computational and communication complexity compared to the similar work.

The most challenging part of our work is to find an efficient way to compute $atan2$ function in a secure two-party computational protocol. We utilize some techniques such as additively blinding a signed angle between two vectors (see Section 3.3.5) as well as Oblivious Transfer protocol (see Section 3.2.1) in order to ensure the location privacy of the participants during the protocol execution.

As illustrated in Section 4.1, our system model consists of a query user (QU) and her friends. For simplicity, we present both protocols in a scenario involving just two parties: Alice (the querier) and Bob. For example, Alice is planning to enjoy her free time after work in one park near her workplace. She wants to know whether Bob is also spending his free time in that park, in order to arrange some outdoor activities together. Therefore, the problem can be stated as follows: Alice specifies an area on the map using a polygon P with vertices $P_i = (x_i, y_i)$, $1 \leq i \leq n$. She can define her polygon either in a clockwise or counterclockwise direction. Bob determines his current location with a single point $M = (a, b)$. Alice wants to check whether Bob is located in her chosen area, without disclosing sensitive information regarding her polygon to Bob, and without getting knowledge of the exact coordinates of M. In the following, we describe both protocols in more detail.

4.3.1 Protocol 1

In our first protocol, all encryptions are performed under Alice's public key, thus the proximity result is only computed by Alice, and Bob does not learn the result. We assume that, before the protocol execution, Alice has published her public key (pk_A). In order to give a general overview of Protocol 1, we summarize it in three main steps as follows.

(a) First, using Alice's encrypted inputs, Bob computes all encrypted det_i and

dot_i (defined in Section 4.2) under Alice's public key.

(b) Next, for each index i, Bob wants to add randomness to the signed angle corresponding to det_i and dot_i (i.e. $atan2\big(det_i, dot_i\big) + \theta'_i$), therefore he computes the encrypted values det'_i and dot'_i in such a way that after decryption, Alice gets: $atan2\big(det'_i, dot'_i\big) = atan2\big(det_i, dot_i\big) + \theta'_i$.

(c) Finally, Bob sends $sum' = \sum_{i=1}^{n} \theta'_i$ along with a list of all encrypted elements (det'_i, dot'_i) to Alice.

Initially, Alice and Bob want to jointly compute Equations 6 and 7 (Step 2 of Algorithm 1 in Section 4.2) for each edge based on their own input, without revealing their input to the other party. Therefore, as detailed in the protocol steps further, Alice encrypts her input for each edge using her public key. She then sends a total of $6n$ ciphertexts to Bob (Step 1). Bob cannot learn anything about Alice's input from these ciphertexts since he does not have Alice's private key to decrypt them. However, by employing the properties of additively homomorphic encryption, he incorporates his input and computes all encryptions of det_i and dot_i (Step 2).

As discussed earlier (Section 3.3.4), each pair of (det_i, dot_i) represents a signed angle θ_i that takes $\overrightarrow{MP_i}$ to $\overrightarrow{MP_{i'}}$, where $i = 1, 2, ..., n$ and $i' = (i{+}1) \mod n$. Alice is interested to calculate the sum of all signed angles θ_i, thus getting $\sum_{i=1}^{n} atan2\big(det_i, dot_i\big)$ as the proximity result. However, if Alice gets the actual values of det_i and dot_i then she can find Bob's exact location by solving two equations (Equations 6 and 7). Therefore, Bob should find a way to mask all actual values of det_i and dot_i, without compromising proximity result accuracy.

To solve the problem, Bob employs the technique suggested in Section 3.3.5 (for each edge) and adds randomness to the signed angle corresponding to det_i and dot_i (i.e. $atan2\big(det_i, dot_i\big) + \theta'_i$). Therefore, he generates two random non-zero numbers r_{x_i} and r_{y_i} such that $\theta'_i = atan2\big(r_{y_i}, r_{x_i}\big)$. According to Equation 5 in Section 3.3.5, Bob knows:

$$atan2\big(det_i, dot_i\big) + atan2\big(r_{y_i}, r_{x_i}\big) = atan2\big(det_i \cdot r_{x_i} + dot_i \cdot r_{y_i}, dot_i \cdot r_{x_i} - det_i \cdot r_{y_i}\big) \quad (8)$$

For convenience, throughout the rest of this work, we denote $(det_i \cdot r_{x_i} + dot_i \cdot r_{y_i})$ and $(dot_i \cdot r_{x_i} - det_i \cdot r_{y_i})$ by det'_i and dot'_i respectively.

As discussed in Section 3.3.5, using Equation 8, the sum of two positive angles could be a negative angle if the output angle crosses π. Similarly, the sum of two negative angles could be a positive angle. To avoid that, Bob should make sure that

the signs of r_{y_i} and det_i are different. As a result, the added random angle θ'_i and $atan2\big(det_i, dot_i\big)$ have different signs.

Since all det_i are encrypted with Alice's public key, Bob needs Alice's help to determine their signs. Therefore, Bob multiplicatively masks all encrypted det_i (Step 3), by generating n random non-zero numbers $r_1, ..., r_n$ (either positive or negative) and computing $E(r_i \cdot det_i)$. The purpose of masking is to avoid Alice from deriving information regarding the sign or the actual value of det_i. Bob then sends all masked det_i to Alice who decrypts them with her private key and returns back signs of all masked det_i (Step 4). Thereby, Bob can decide about the sign of r_{y_i} using the following comparison (Step 5.a). If r_i and $E(r_i \cdot det_i)$ have the same sign, it means $det_i \geq 0$, then r_{y_i} should be negative. Likewise, if r_i and $E(r_i \cdot det_i)$ do not have the same sign, it means $det_i < 0$, then r_{y_i} should be positive.

Remark. *At this point, signs of all det_i are disclosed to Bob. However, as will be discussed in detail in Chapter 5, the leaked information reveals some minor information about Alice's polygon to Bob in special cases, but does not violate the privacy of the whole scheme.*

Next, Bob wants to compute $\big(det_i \cdot r_{x_i} + dot_i \cdot r_{y_i}\big)$ and $\big(dot_i \cdot r_{x_i} - det_i \cdot r_{y_i}\big)$ in Equation 8. Accordingly, he employs the properties of additively homomorphic encryption and incorporates r_{x_i} and r_{y_i} in the encrypted values det_i and dot_i, thus getting the encryptions of det'_i and dot'_i (Step 5.b). He then sends $sum' = \sum_{i=1}^{n} atan2\big(r_{y_i}, r_{x_i}\big)$ along with a list of all encrypted elements (det'_i, dot'_i) to Alice who decrypts them with her private key and computes the proximity result as follows:

$$\theta = \big(\sum_{i=1}^{n} atan2\big(det'_i, dot'_i\big)\big) - sum'$$

The detailed protocol is described below.

Protocol 1 Privacy-Preserving Scheme for the Point Inclusion Problem

---

Input: Alice has a polygon P with vertices P_i, $1 \leq i \leq n$; her public/private keypair. Bob has a point $M = (a, b)$; Alice's public key pk_A.

Output: Alice knows whether M lies inside P or not.

Protocol steps:

1. Alice computes $\big(\langle x_i x_{i'} + y_i y_{i'} \rangle, \langle -x_i - x_{i'} \rangle, \langle -y_i - y_{i'} \rangle, \langle x_{i'} - x_i \rangle, \langle y_i - y_{i'} \rangle, \langle x_i y_{i'} - x_{i'} y_i \rangle\big)$ for each index i and sends these ciphertexts to Bob, where $i = 1, 2, ..., n$, $i' = (i + 1) \mod n$.

2. For each index i, where $i = 1, 2, ..., n,\ \ i' = (i+1) \bmod n$, Bob using Alice's public key (pk_A) computes:

$$\big\langle x_i x_{i'} + y_i y_{i'} \big\rangle \cdot \big\langle -x_i - x_{i'} \big\rangle^a \cdot \big\langle -y_i - y_{i'} \big\rangle^b \cdot \big\langle a^2 + b^2 \big\rangle$$
$$= \big\langle x_i x_{i'} + y_i y_{i'} - a(x_i + x_{i'}) - b(y_i + y_{i'}) + a^2 + b^2 \big\rangle = \big\langle v_i \cdot v_{i'} \big\rangle = \big\langle dot_i \big\rangle_{pk_A}$$
$$\big\langle x_i y_{i'} - x_{i'} y_i \big\rangle \cdot \big\langle x_{i'} - x_i \big\rangle^b \cdot \big\langle y_i - y_{i'} \big\rangle^a$$
$$= \big\langle x_i y_{i'} - x_{i'} y_i + b(x_{i'} - x_i) + a(y_i - y_{i'} \big\rangle = \big\langle det(v_i, v_{i'}) \big\rangle = \big\langle det_i \big\rangle_{pk_A}$$

3. Bob generates n random values $r_1, ..., r_n \in I_1 - \{0\}$ (blinding factor domain is discussed in Section 4.5) for masking the determinants, and computes:

$$d_i = (\big\langle det_i \big\rangle_{pk_A})^{r_i} = \big\langle r_i \cdot det_i \big\rangle_{pk_A} \qquad 1 \le i \le n$$

   Next, he sends $(d_1, .., d_n)$ back to Alice.

4. Alice decrypts the list obtained from Bob using her private key; she then determines the sign of decrypted elements for $1 \le i \le n$:

$$\text{If } \big(D(\big\langle d_i \big\rangle_{pk_A}) \ge 0\big) \text{ then } sign_i = +1, \text{ otherwise } sign_i = -1.$$

   Alice sends $(sign_1, ..., sign_n)$ back to Bob.

5. For each $i = 1, ..., n$, Bob conducts the following sub-steps:

   (a) Considering the following condition, Bob generates two random numbers $r_{x_i},\ r_{y_i} \in I_2 - \{0\}$ (blinding factor domain is discussed in Section 4.5).

   If r_i and $sign_i$ have the same sign then $r_{y_i} < 0$, otherwise $r_{y_i} > 0$.

   (b) Using Equation 8, Bob calculates:

$$\big\langle det_i \big\rangle^{r_{x_i}} \cdot \big\langle dot_i \big\rangle^{r_{y_i}} = \big\langle det_i \cdot r_{x_i} + dot_i \cdot r_{y_i} \big\rangle_{pk_A} = \big\langle det'_i \big\rangle_{pk_A}$$
$$\big\langle dot_i \big\rangle^{r_{x_i}} \cdot \big\langle det_i \big\rangle^{-r_{y_i}} = \big\langle dot_i \cdot r_{x_i} - det_i \cdot r_{y_i} \big\rangle_{pk_A} = \big\langle dot'_i \big\rangle_{pk_A}$$

6. Bob sends $\big((\big\langle det'_1 \big\rangle, \big\langle dot'_1 \big\rangle), ..., (\big\langle det'_n \big\rangle, \big\langle dot'_n \big\rangle)\big)$ and $sum' = \sum_{i=1}^{n} atan2\big(r_{y_i}, r_{x_i}\big)$ to Alice.

7. Alice decrypts all encrypted received pairs using her private key; she then computes:

$$\theta = \big(\sum_{i=1}^{n} atan2\big(det'_i, dot'_i\big)\big) - sum'$$

If $\theta = \pm 360°$, the point M lies inside the polygon P. Otherwise, if $\theta = 0°$, M is outside the polygon P. As explained in Section 4.2, if point M coincides with one of the vertices of P, the result is undefined.

In the following, Protocol 1 is summarized.

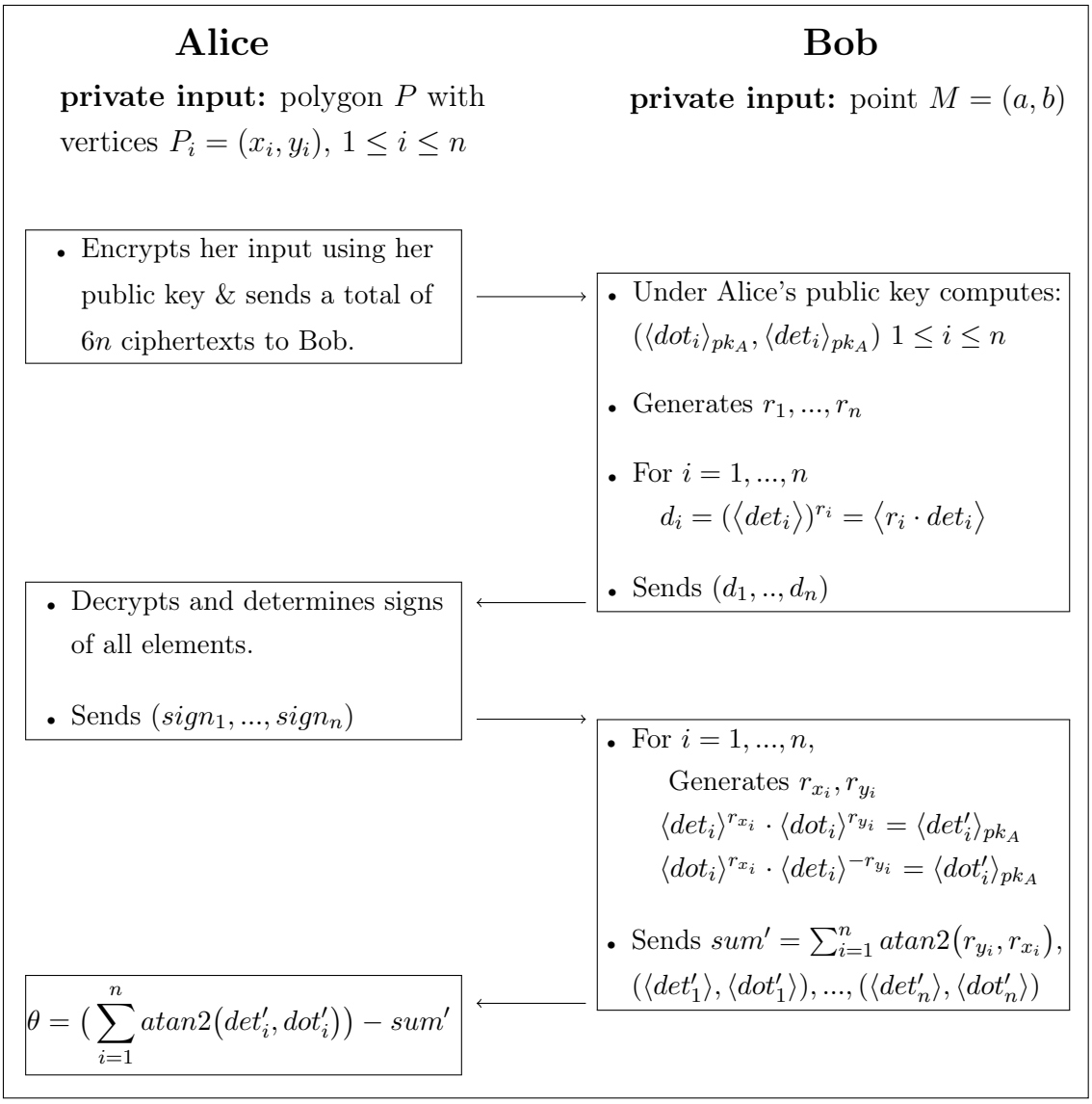


Figure 12: Protocol 1

4.3.2 Protocol 2

The first proposed protocol reveals signs of all det_i to Bob. As will be discussed in detail in Chapter 5, the leaked information reveals some minor information about Alice's polygon to Bob in some scenarios. In this section, we propose another protocol trying not to reveal any additional information to the other party that might cause privacy violations. Our proposed protocol is inspired by secure two-party scalar product protocol presented by Atallah and Du [AD01] which already is discussed in detail in Section 2.3.

Unlike Protocol 1, this protocol performs all encryptions under Bob's public key. We assume that, before the protocol execution, Bob has published his public key (pk_B). Similar to the previous solution, initially, Alice and Bob want to jointly compute Equations 6 and 7 (Step 2 of Algorithm 1 in Section 4.2) for each edge based on their own input. In the following, we describe the detailed protocol which is presented later in this section. At first, Alice starts the protocol by sending a request to Bob asking him to send his encrypted coordinates (Step 2). In the next step, Bob encrypts his inputs using his public key; he then sends a total of three ciphertexts to Alice (Step 3). Next, Alice utilizes the homomorphic encryption properties in order to incorporate her own input; thereby she computes all encryptions of det_i and dot_i (Step 5). However, if Bob gets all actual values of det_i and dot_i, then he can conclude some information about Alice's polygon by solving a system of $2n$ nonlinear equations (this system of equations are discussed in Section 5.1.2). Therefore, Alice should find a way to prevent Bob from getting access to actual values of det_i and dot_i, without compromising proximity result accuracy.

Consider the following solution: For each index i, where $i = 1, ..., n$, Alice sends two lists to Bob. The first list contains p encrypted numbers, only one of which is $\langle dot_i \rangle_{pk_B}$ and the rest are random dot products encrypted with Bob's public key. Similarly, the second list contains p encrypted numbers including $\langle det_i \rangle_{pk_B}$ and $p-1$ arbitrary determinants. After decrypting the two lists, Bob computes p^2 smallest signed angles for all possible combinations of determinants and dot products using Equation 3 (in Section 3.3.4); then he adds θ'_i to all computed angles, where θ'_i is a random real number and $\sum_{i=1}^{n} \theta'_i = v$. The purpose of θ'_i is to prevent Alice from getting the exact signed angle between the two vectors. At the end Alice employs the 1-out-of-p^2 oblivious transfer protocol to get $Z_i = atan2\big(det_i, dot_i\big) + \theta'_i$ from Bob (see Figure 13). Because of the way oblivious transfer protocol works, Alice can securely get the computed angle corresponding to det_i and dot_i, while Bob could

not learn which one Alice has selected.

Certainly, for each index i, there is 1 out of p^2 possibility that Bob can guess the correct values of det_i and dot_i. However, many guesses could lead to self-intersecting polygons or improbable shapes. As a result, some of the guesses lead to more probable polygons than others. Therefore, if we choose p and n to be large enough, Bob's chance of inferring sensitive information about Alice's polygon is negligible.

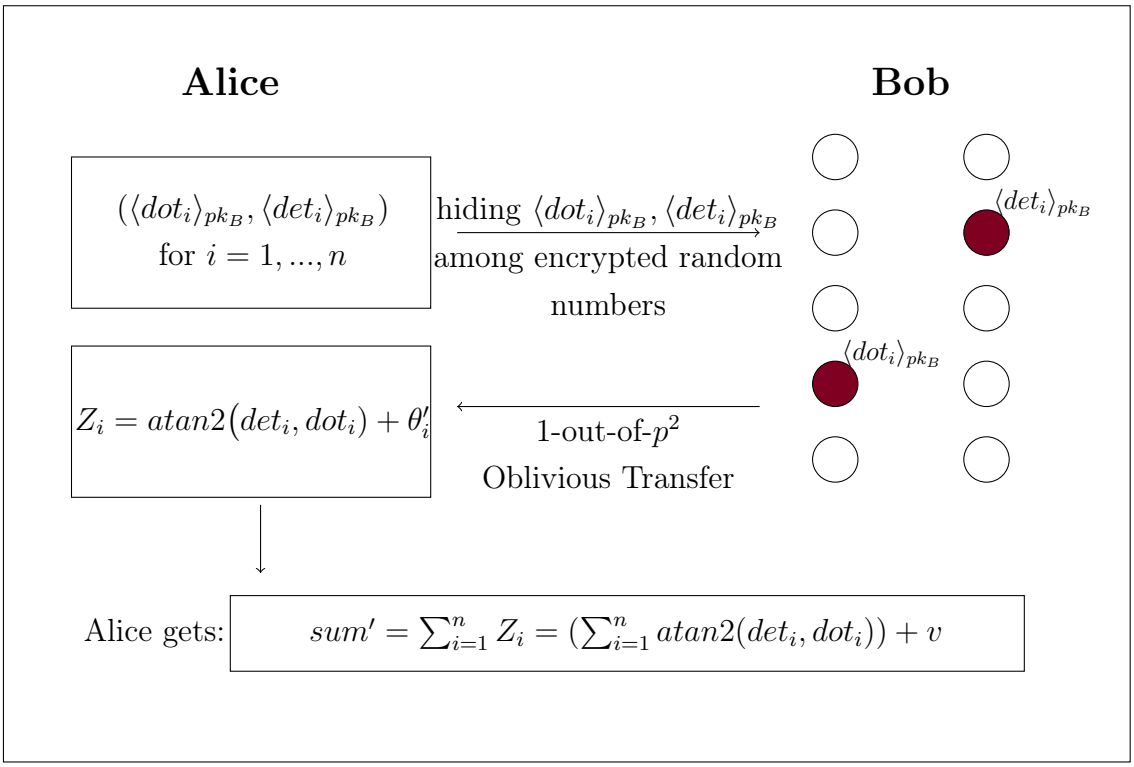


Figure 13: Using the oblivious transfer protocol to present a privacy-preserving scheme for the point inclusion problem.

Privacy guarantees: Protocol 2 provides privacy guarantees as follows:

- Alice learns only the proximity result. She cannot get Bob's exact location by analyzing the information exchanged during the protocol execution.
- Bob does not learn the proximity result. Furthermore, during protocol execution, the number of edges in Alice's polygon, as well as all det_i and dot_i hidden among fake values are revealed to him. If Bob chooses to guess, his chance of inferring sensitive information about Alice's polygon is negligible if we choose p and n to be large enough.

The detailed protocol is described below and the OT table created at Bob's side is presented in Table 1.

Table 1: OT table at Bob's side

Key	Values
1	$Z_{i,1} = atan2\big(T_1, H_1\big) + \theta'_i$
2	$Z_{i,2} = atan2\big(T_1, H_2\big) + \theta'_i$
$\vdots$	$\vdots$
p	$Z_{i,p} = atan2\big(T_1, H_p\big) + \theta'_i$
$p+1$	$Z_{i,p+1} = atan2\big(T_2, H_1\big) + \theta'_i$
$\vdots$	$\vdots$
p^2-1	$Z_{i,p^2-1} = atan2\big(T_p, H_{p-1}\big) + \theta'_i$
p^2	$Z_{i,p^2} = atan2\big(T_p, H_p\big) + \theta'_i$

Protocol 2 Privacy-Preserving Scheme for Point Inclusion Problem Using OT

Input: Bob has a point $M = (a, b)$; his public/private keypair. Alice has a polygon P with vertices P_i, $1 \leq i \leq n$; Bob's public key pk_B.

Output: Alice knows whether M lies inside P or not.

Protocol steps:

1. Alice and Bob agree on number p, such that p is large enough.
2. Alice sends a request to Bob asking him to send his encrypted coordinates.
3. Bob computes $\big(\langle a \rangle_{pk_B}, \langle b \rangle_{pk_B}, \langle a^2 + b^2 \rangle_{pk_B}\big)$ and sends these ciphertexts to Alice.
4. Bob generates n uniformly distributed random angles $\theta'_1, ..., \theta'_n \in \mathbb{R}$ (real numbers) such that $v = \sum_{i=1}^{n} \theta'_i$.
5. For each index i, where $i = 1, 2, ..., n$, $i' = (i+1) \mod n$, Alice using Bob's public key (pk_B) computes:

$$\langle x_i x_{i'} + y_i y_{i'} \rangle \cdot \langle a \rangle^{-(x_i + x_{i'})} \cdot \langle b \rangle^{-(y_i + y_{i'})} \cdot \langle a^2 + b^2 \rangle$$

$$= \langle x_i x_{i'} + y_i y_{i'} - a(x_i + x_{i'}) - b(y_i + y_{i'}) + a^2 + b^2 \rangle = \langle v_i \cdot v_{i'} \rangle = \langle dot_i \rangle_{pk_B}$$

$$\langle x_i y_{i'} - x_{i'} y_i \rangle \cdot \langle b \rangle^{(x_{i'} - x_i)} \cdot \langle a \rangle^{(y_i - y_{i'})}$$

$$= \langle x_i y_{i'} - x_{i'} y_i + b(x_{i'} - x_i) + a(y_i - y_{i'} \rangle = \langle det(v_i, v_{i'}) \rangle = \langle det_i \rangle_{pk_B}$$

6. For each $i = 1, ..., n$, Alice and Bob do the following sub-steps:

   (a) Alice generates two secret random numbers k_1 and k_2 in the range $[1, p]$.

   (b) Alice sends two lists to Bob: $H = (H_1, ..., H_p)$ and $T = (T_1, ..., T_p)$, where $H_{k_1} = \langle dot_i \rangle_{pk_B}$ and $T_{k_2} = \langle det_i \rangle_{pk_B}$, and the rest of H_j's and T_j's are random dot products and determinants encrypted with Bob's public key respectively. Since k_1 and k_2 are secret numbers known only to Alice, the position of $\langle dot_i \rangle_{pk_B}$ and $\langle det_i \rangle_{pk_B}$ are unknown to Bob.

   (c) Bob decrypts two lists H and T using his private key, thus getting $D_B(H)$ and $D_B(T)$, then creates an OT table with p^2 entries shown in Table 1, such that:

$$Z_{i,(j-1)p+k} = atan2\big(T_j, H_k\big) + \theta'_i \qquad 1 \le j \le p \quad 1 \le k \le p$$

   (d) Alice employs the 1-out-of-p^2 Oblivious Transfer protocol and gets $Z_i = Z_{i,(k_2-1)p+k_1} = atan2\big(det_i, dot_i\big) + \theta'_i$, while Bob does not learn anything about $(k_2 - 1)p + k_1$.

7. Bob sends $v = \sum_{i=1}^{n} \theta'_i$ to Alice.

8. Alice calculates:

$$\theta = \sum_{i=1}^{n} Z_i - v = \sum_{i=1}^{n} atan2\big(det_i, dot_i\big)$$

   If $\theta = \pm 360°$, the point M lies inside the polygon P. Otherwise, if $\theta = 0°$, M is outside the polygon P. As explained in Section 4.2, if point M coincides with one of the vertices of P, the result is undefined.

4.4 Hiding the Number of Vertices

Both proposed protocols reveal the number of edges in Alice's polygon to Bob. In this section, we propose a hiding strategy that enables Alice to protect the number of edges in her polygon against disclosure to Bob during protocol execution. The key idea is the following. Before executing the protocol, in one preprocessing phase, Alice increases the number of vertices in her polygon by adding some random points on each edge of the polygon (see Figure 14). Therefore, she prevents Bob from learning the number of edges in her chosen polygon. Moreover, the added vertices do not affect the accuracy of the result. If Bob chooses to guess, his chance of guessing the correct number of edges is 1 out of $n + m$, where m is the number of added vertices.

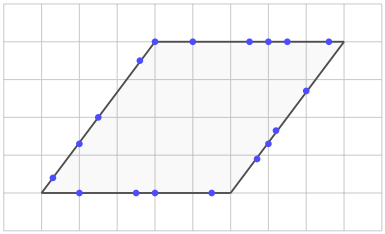


Figure 14: Adding random points on each edge of P.

4.5 Random Blinding Factor Domain

In this section, we take a closer look at how to choose the random blinding factor domain in Protocol 1 which will not cause an overflow (a positive number suddenly becomes a negative number or vice-versa). As discussed in Section 3.2.4, we use the top half of the message space for representing negative numbers. For security considerations, all randomly chosen blinding factors in Protocol 1 must be either positive or negative. The representation of negative numbers follows the standard two's complement representation, and thus the range of supported numbers for δ bits is from $-2^{\delta-1}$ to $(2^{\delta-1} - 1)$. Besides, we assume that all random variables are uniformly chosen as described with more details in the following.

In Step 3 of Protocol 1, Bob masks the real values of determinants by multiplying them in random blinding factors r_i (i.e, $\langle r_i \cdot det_i \rangle$). Moreover, in Step 5 of Protocol 1, Bob generates random blinding factors r_{x_i} and r_{y_i}, then he computes the blinded

values of determinants and dot products as follows.

$$\begin{cases} \langle det_i \cdot r_{x_i} + dot_i \cdot r_{y_i} \rangle = \langle det'_i \rangle \\ \langle dot_i \cdot r_{x_i} - det_i \cdot r_{y_i} \rangle = \langle dot'_i \rangle \end{cases}$$

Let us assume that the maximum value of det_i and dot_i is represented with n'. As shown in Figure 15, the blinding factor r_i should be chosen in a way that does not cause det_i to overflow, which would lead to an incorrect result. Moreover, it should be large enough to statistically hide the magnitudes of det_i from Alice. As a result, the following condition should be satisfied.

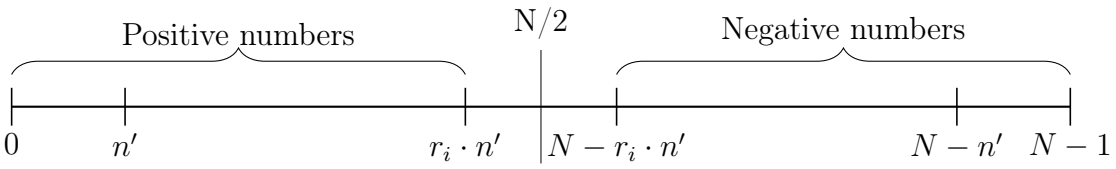


Figure 15: Random blinding factor domain in Protocol 1.

$$r_i \cdot n' < N - r_i \cdot n' \Leftrightarrow r_i < \tfrac{N}{2n'} \Rightarrow$$
$$r_i \text{ is in order of } \alpha = \log N - (\log n' + 1) \text{ bits.}$$
$$r_i \in I_1 - \{0\} \text{ where } I_1 = \{-2^{\alpha-1}, ..., (2^{\alpha-1} - 1)\}$$

Since binary addition requires a carry bit, the length of r_{x_i} and r_{y_i} should be at least one bit less than r_i. Therefore, it can be concluded that:

$$r_{x_i}, r_{y_i} < \tfrac{N}{4n'} \Rightarrow$$
$$r_{x_i}, r_{y_i} \text{ are in order of } \beta = \log N - (\log n' + 2) \text{ bits.}$$
$$r_{x_i}, r_{y_i} \in I_2 - \{0\} \text{ where } I_2 = \{-2^{\beta-1}, ..., (2^{\beta-1} - 1)\}$$

Example 4.1. *Let us set* $\log N = 1024$ *and* $\log n' = 64$. *In this case,* r_i *is in order of* 959 *bits, sufficiently large to obtain a strong degree of protection through random blinding, and the range of supported numbers is from* -2^{958} *to* $(2^{958} - 1)$ *(excluding 0). Similarly,* r_{x_i} *and* r_{y_i} *are in order of* 958 *bits and the range of supported numbers is from* -2^{957} *to* $(2^{957} - 1)$ *(excluding 0).*

Remark. *As explained in Section 3.4, during the protocol execution, floating point values are converted to fixed precision. Thus, if we choose* $e = 6$ *(large enough to get the accurate result) then* 64 *bits are sufficient for representing* det_i *and* dot_i.

5 Evaluation

This chapter assesses the security and performance aspects of two suggested privacy-preserving protocols for the point-inclusion problem. The security of the proposed schemes is evaluated with respect to the security requirements presented in Section 4.1.2. In addition, the performance of the suggested protocols is evaluated in terms of computation complexity and communication overhead.

5.1 Security Analysis

In this section, we take a closer look at the security of two protocols. In our schemes, all encryptions are performed using Paillier cryptosystem, a homomorphic encryption scheme, which is semantically secure (defined in Section 3.2.3). To demonstrate the security of the protocols, the view of each party is scrutinized using its inputs and outputs. Then we investigate how our proposed privacy-preserving schemes preserve the privacy of the participants against a curious Alice and a curious Bob.

5.1.1 Security Analysis of Protocol 1

Bob's view of the protocol

Bob's input is a fixed point $M = (a, b)$; he has no output. In Step 1, Bob receives a total of $6n$ ciphertexts from Alice. The semantic security of Paillier cryptosystem prevents Bob from deriving any information regarding Alice's real input from these ciphertexts. Furthermore, in Step 4, signs of all determinants are revealed to Bob. Therefore, a curious Bob might attempt to deduce some information regarding Alice's polygon (e.g, shape, size and location) from the obtained signs. In the following, a comprehensive analysis of different cases that might happen is provided.

Consider that the disclosed signs to Bob are as follows: $Sign(det) = \{+ + - + -\}$. As discussed in Section 3.3.3, geometric interpretation of a positive determinant sign indicates that the rotation angle between two vectors, starting from Bob's location to two consecutive vertices of the polygon, is in a counter-clockwise direction, while a negative sign indicates that the rotation angle is in a clockwise direction. Two possible selected directions for listing the vertices of P are either clockwise or counter-clockwise, which is a secret parameter known only to Alice. Another unknown parameter to Bob is the proximity result. In our scheme, the proximity result

is computed only by the querier (Alice). If Bob learns the result, it might provide incentives for Bob to collude with other participating entities in order to infer some sensitive information about Alice's query. As a result, Alice's query would be divulged. Figure 16 and 17 depict some possible polygons that might be drawn by Bob in his attempt to learn some information about Alice's polygon according to his location at query time. In the former, the counter-clockwise direction is chosen for drawing the polygons, and in the latter the clockwise direction. It can be seen that the drawn polygons have different shapes, with arbitrary sizes, located in different locations. Consequently, if Bob chooses to guess, his chance of guessing the exact P seems very small.

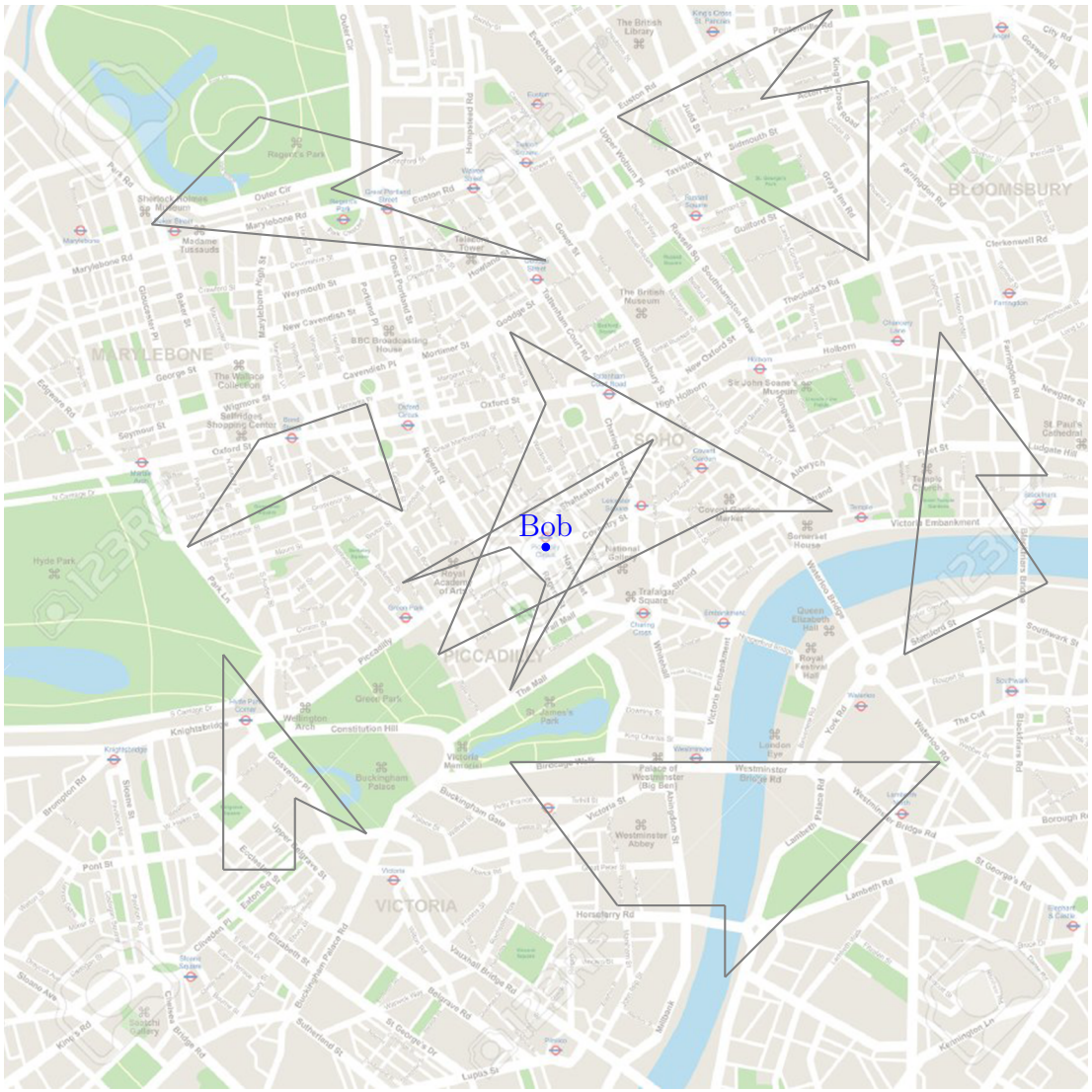


Figure 16: Drawn polygons in Counterclockwise direction.

Figure 17: Drawn polygons in Clockwise direction.

There are two special cases that need to be considered separately including cases where all obtained signs are either positive or negative. Therefore, the information that is concluded by Bob is as follows. In both cases, he learns that he encloses by Alice's polygon. Moreover, he learns about the chosen direction for naming the vertices of P. If all signs are positive, the polygon's vertices are named in the anticlockwise direction. Likewise, the clockwise direction is chosen for naming the polygon's vertices, if all signs are negative. Figure 18 shows some possible polygons that might be guesstimate by Bob where all obtained signs are either positive or negative. Due to the variety of drawn polygons, the probability of deriving information regarding the exact shape and size of Alice's polygon, in reality, is very low (almost near zero).

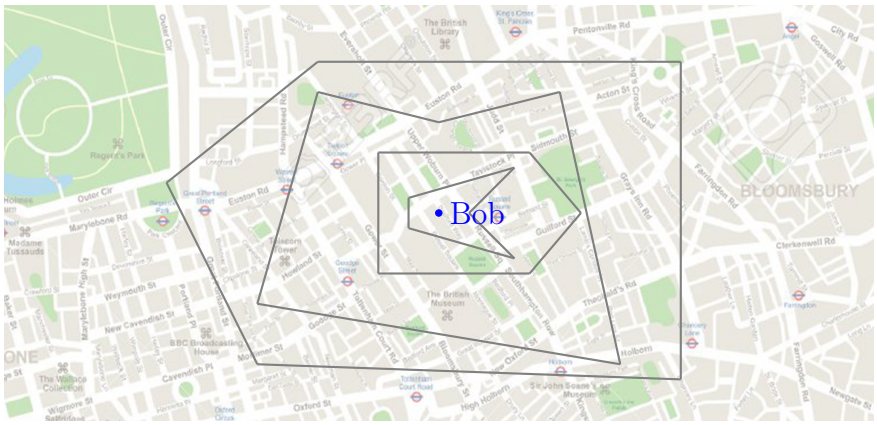


Figure 18: Samples of selected areas by Bob where all obtained signs are either positive or negative.

It is noteworthy that in Protocol 1, we assume that the party who owns the polygon (Alice) is the querier. However, with a slight modification to our first protocol, it is possible for the other party who owns the fixed point (Bob) to initiate a proximity detection query. One such scenario is demonstrated in [SS05] when a user wants to privately determine whether he is located inside the sensing area of a pervasive sensor network or not. In this case, our proposed privacy-preserving scheme cannot preserve the privacy of Alice's polygon against *chosen-point attack* performed by Bob. In the following, more detail and elaboration about this attack is provided.

At first, Bob sends a proximity detection query to Alice, and he obtains a list (e.g, S_1) containing signs of all determinants with respect to his location at the query time. In the next step, he changes his location a bit. Then he sends another query to Alice which results in obtaining another list of signs (e.g, S_2). Since he has moved a bit, he can determine if he has crossed one edge of the polygon (or the line determined by one edge of the polygon). In the case of crossing, by comparing S_2 to S_1, one sign has changed from positive to negative or vice-versa. Therefore, he can mark the crossing point as one critical point. By repeating the above steps, Bob gets a list of critical points. Therefore, by determining which points are collinear, he can connect them together and obtains sensitive information about Alice's polygon including shape, size, and its location. In such a scenario, one possible solution for Alice is to accept only limited number of queries originated from Bob. The chosen-point attack against Alice's polygon is depicted in Figure 19.

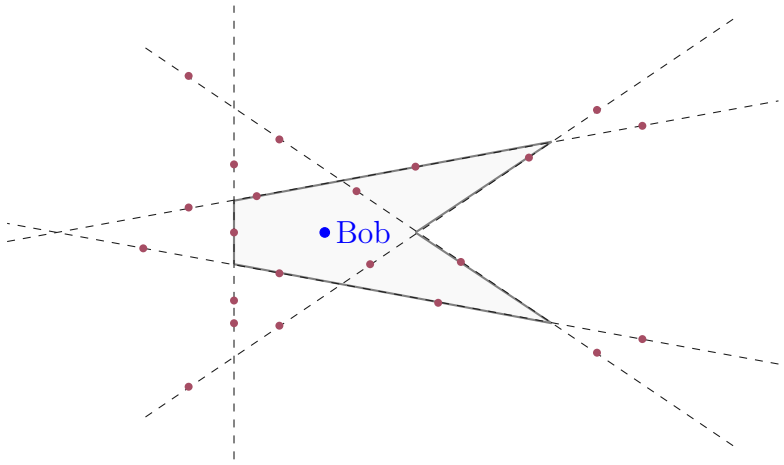


Figure 19: Chosen-point attack against Alice's polygon originated from Bob. Red dots are such location of Bob that one of the signs has swapped.

Alice's view of the protocol

Alice's input is n vertices and her output is $\theta = \left(\sum_{i=1}^{n} atan2\left(det'_i, dot'_i\right)\right) - sum'$. In step 3, Alice receives a list of n randomized determinants (i.e, $E(r_i \cdot det_i)$) from Bob. The randomization resulted from $r_i \in I_1 - \{0\}$ for $1 \leq i \leq n$ (see Section 4.5), which can statistically hide det_i from Alice. At the same time, r_i will not cause det_i to overflow, which would lead to an incorrect result. Here, we have a minor note. Since the blinding factors are non-zero, if Alice receives one zero among all masked values, she learns that the related determinant would be zero. From the geometric perspective, if two vectors point in either the same direction or exact opposite directions then the corresponding determinant is zero (as shown in Figure 20)

(a) If two vectors point in exact opposite directions then $det(\vec{v}, \vec{w}) = 0$ and $dot(\vec{v}, \vec{w}) < 0$.

(b) If two vectors point in the same direction then $det(\vec{v}, \vec{w}) = 0$, $dot(\vec{v}, \vec{w}) > 0$.

Figure 20

Consequently, she concludes that Bob is located either on one of the polygon edges or on the line determined by one edge of the polygon. If final output determines that Bob is inside the polygon, then Alice learns Bob is located on one of the

polygon edges. One possible solution for Bob is to permute the list of all masked determinants before sending them to Alice. The purpose of permutation is to reduce Alice's chance of guessing the correct edge (or a line segment) on which Bob might be located (as demonstrated in Figure 21). It is worth noting that, in reality, the chance of being located on the edge of one chosen geographic area is very small. By improving the granularity of the location measurement, the chance of being located on the edge is further decreased. Note that the granularity of the measurement could even be made artificially more fine-grained by including random digits to achieve accuracy for millimeter (or even micrometer) range.

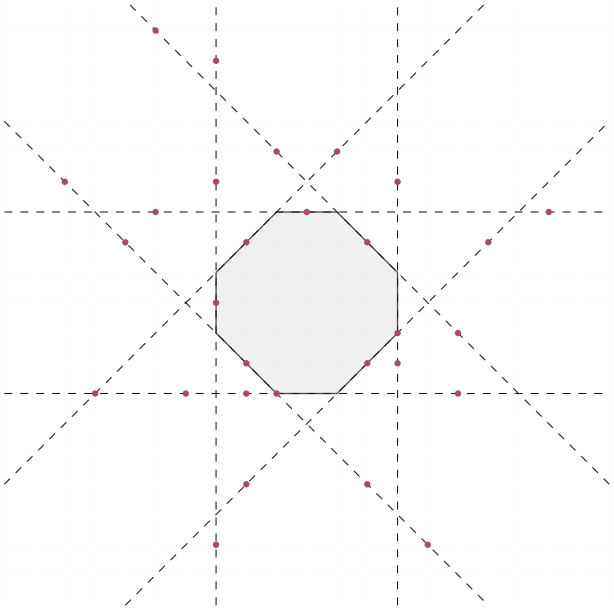


Figure 21: Red points show some locations that Bob might be located in when one determinant is zero.

Furthermore, in Step 6, Alice receives $(\langle det'_i \rangle, \langle dot'_i \rangle)$ for $i = 1, ..., n$, and $sum' = \sum_{i=1}^{n} atan2(r_{y_i}, r_{x_i})$ from Bob. For convenience of discussion, let θ_i denote the exact signed angle that takes $\overrightarrow{MP_i}$ to $\overrightarrow{MP_{i+1}}$.

As we know:

$$\begin{cases} det_i \cdot r_{x_i} + dot_i \cdot r_{y_i} = det'_i \\ dot_i \cdot r_{x_i} - det_i \cdot r_{y_i} = dot'_i \end{cases}$$

It follows that:

$$(r_{x_i} - r_{y_i})det_i + (r_{x_i} + r_{y_i})dot_i = det'_i + dot'_i$$

where r_{x_i}, $r_{y_i} \in I_2 - \{0\}$ (see Section 4.5). The purpose of r_{x_i} and r_{y_i} are to statistically hide the real det_i and dot_i from Alice. Moreover, they will add randomness to θ_i, which will prevent Alice from guessing the location of Bob. That is

$$\underbrace{atan2\big(det_i, dot_i\big)}_{\theta_i} + \underbrace{atan2\big(r_{y_i}, r_{x_i}\big)}_{\theta'_i} = \underbrace{atan2\big(det'_i, dot'_i\big)}_{\theta_i+\theta'_i}$$

where $\theta_i \in (-\pi, \pi]$, $\theta_i + \theta'_i \in (-\pi, \pi)$ and $\theta'_i \in (-\pi, \pi) - \{-\frac{\pi}{2}, 0, \frac{\pi}{2}\}$. At the same time, θ'_i will not cause θ_i to overflow, which would lead to an incorrect result.

There is one special case that needs to be considered more carefully. If the value of $\theta_i + \theta'_i$ is close to π or $-\pi$, then Alice might conclude some information about Bob's location. Let us clarify the situation with an example. Assume that $\theta_i + \theta'_i = 175°$. Two possible estimations for the values of θ_i and θ'_i are as follows.

1. The exact signed angle $\theta_i \in (175°, 180°]$, and the added random angle $\theta'_i \in [-5°, 0)$. In this case, Bob lies near the ith edge of Alice's polygon.

2. The exact signed angle $\theta_i \in (-5°, 0]$, and the added random angle $\theta'_i \in [175°, 180°)$. In this case, Bob is located far away from the ith edge of Alice's polygon.

In some cases, it is even possible to rule out one of the above options. For example, let us consider the following situation. If the proximity result shows that Bob is located inside Alice's polygon, by considering the shape and size of the polygon, Alice might figure out that the chance of Bob locating inside her chosen area, together with θ_i near to zero is almost impossible. Therefore, she concludes that Bob is located near the ith edge of her polygon.

One possible way to mitigate this problem is to permute the list of all pairs $(\langle det'_i \rangle, \langle dot'_i \rangle)$ for $i = 1, ..., n$, before sending them to Alice. In this case, Alice's chance of guessing the correct edge is 1 out of n. Another possible mitigation is to provide Bob with one security parameter (before executing the protocol). Therefore, he can supervise the chosen random numbers r_{x_i} and r_{y_i}, such that the value of $\theta'_i = atan2\big(r_{y_i}, r_{x_i}\big)$ would not be close to π, $-\pi$ and 0. Since the security parameter is known only to Bob, Alice cannot derive sensitive information that might violate the privacy of Bob's location. However, the solution suggested above deserves further investigation.

5.1.2 Security Analysis of Protocol 2

Alice's view of the protocol

Alice's input consists of n vertices and her output is $\theta = \sum_{i=1}^{n} Z_i - v$. In Step 3, Alice receives a total of 3 ciphertexts from Bob (encrypted with Bob's public key). The semantic security of Paillier cryptosystem prevents Alice from deriving any information regarding Bob's real input from these ciphertexts. Furthermore, in Step 6(d) and 7, Alice gets $Z_i = atan2\big(det_i, dot_i\big) + \theta'_i$, where $i = 1, ..., n$, and $v = \sum_{i=1}^{n} \theta'_i$. The purpose of θ'_i is to add randomness to the signed angle between the edges $\overline{MP_i}$ and $\overline{MP_{i+1}}$, where M is Bob's location. Since θ'_i is a random element of the set $\mathbb{R}$ (real numbers), the random masking provides encryption that resembles the *one-time pad* (OTP), which is proved to provide perfect secrecy [Sha49]. Therefore, Alice's chance of guessing the exact location of Bob is very small.

Bob's view of the protocol

Bob's input is a fixed point $M = (a, b)$; he has no output. In Step 6(b), Bob receives a total of $2n$ lists from Alice; each list contains p encrypted elements (one real determinant or dot product hidden among $p-1$ random elements). Therefore, there is 1 out of p possibility that Bob might guess one correct det_i or dot_i. However, many random combinations could lead to self-intersecting polygons or improbable shapes. If Bob chooses to guess, his chance of inferring sensitive information about Alice's polygon is negligible if we choose p and n to be large enough.

In a case of guessing all exact values, then Bob should solve a system of nonlinear equations as explained in the following.

Let us assume that Alice's polygon has only 3 vertices. First, Bob defines vectors $\vec{v_1}$, $\vec{v_2}$, $\vec{v_3}$ as follows.

$$\vec{v_1} = (x_1 - a, y_1 - b),\ \vec{v_2} = (x_2 - a, y_2 - b),\ \vec{v_3} = (x_3 - a, y_3 - b)$$

Then he builds a system of equations as:

$$\vec{v_1} \cdot \vec{v_2} = dot_1,\ \vec{v_2} \cdot \vec{v_3} = dot_2,\ \vec{v_3} \cdot \vec{v_1} = dot_3,$$
$$\det(\vec{v_1}, \vec{v_2}) = det_1,\ \det(\vec{v_2}, \vec{v_3}) = det_2,\ \det(\vec{v_3}, \vec{v_1}) = det_3$$

As defined in Section 3.3.2 and 3.3.3, geometric definition of the dot product and determinant can be determined as:

$$|\vec{v_1}|\,|\vec{v_2}|\cos\vartheta_{12} = dot_1,\ |\vec{v_2}|\,|\vec{v_3}|\cos\vartheta_{23} = dot_2,\ |\vec{v_3}|\,|\vec{v_1}|\cos\vartheta_{31} = dot_3,$$
$$|\vec{v_1}|\,|\vec{v_2}|\sin\vartheta_{12} = det_1,\ |\vec{v_2}|\,|\vec{v_3}|\sin\vartheta_{23} = det_2,\ |\vec{v_3}|\,|\vec{v_1}|\sin\vartheta_{31} = det_3$$

Where ϑ_{ij} is the angle between vectors $\vec{v_i}$ and $\vec{v_j}$. By squaring the above equations and using $\sin^2 t + \cos^2 t = 1$, Bob gets:

$$|\vec{v_1}|\,|\vec{v_2}| = c_1,\ |\vec{v_2}|\,|\vec{v_3}| = c_2,\ |\vec{v_3}|\,|\vec{v_1}| = c_3$$

Where $c_i = \sqrt{dot_i^2 + det_i^2}$. Thus, he can find the length of $|\vec{v_i}|$, by solving the following equations.

$$|\vec{v_1}| = \sqrt{\frac{c_1 c_3}{c_2}},\ |\vec{v_2}| = \sqrt{\frac{c_1 c_2}{c_3}},\ |\vec{v_3}| = \sqrt{\frac{c_2 c_3}{c_1}}.$$

As a result, by solving the above equations[1], Bob gets sensitive information regarding the location, shape, and size of P. It is noteworthy that solving the above equations even with having some exact values of det_i and dot_i, might still reveal sensitive information about Alice's polygon to Bob.

[1]We thank user @Ennar in the mathematics community of https://stackexchange.com/ who helped us to solve this system of equations.

5.2 Performance Analysis

In this section, we evaluate the performance of our proposed protocols regarding computation complexity, round complexity and communication overhead. The computation complexity is expressed in terms of the number of operations required to compute the result. The communication overhead is expressed in terms of the amount of data exchanged between two parties during the protocol execution, while the round complexity is the number of communication rounds.

Let us assume that N is modulus in Paillier's homomorphic encryption scheme, and n is the number of vertices in Alice's polygon. In the following, we analyze the performance of the two protocols.

5.2.1 Computation Complexity

Table 2 shows a list of notations representing operation costs used throughout this section. Small exponents are involved when a party multiplies its plaintext into one ciphertext, while large exponents are involved in multiplicative masking operations using randomly chosen large numbers. For simplicity, the cost of addition as well as choosing random numbers are neglected.

Table 2: Notations of operation costs

Encryption	T_e
Decryption	T_d
Inversion	T_i
Modular exponentiation (small exponent)	T_{se}
Modular exponentiation (large exponent)	T_{le}
Modular multiplication	T_m
Inverse tangent	T_{atan2}
Computation cost of $\mathrm{OT}^1_{p^2}$ for Alice	T_{OT_A}
Computation cost of $\mathrm{OT}^1_{p^2}$ for Bob	T_{OT_B}

Remark. *We assume that in reality about half of the modular exponentiations will be performed with a negative exponent since the inputs of both parties, as well as the chosen random numbers, could be negative.*

Protocol 1: In Step 1 of Protocol 1, Alice does $6n$ encryptions. Next, in Steps 2

and 3, Bob conducts 1 encryption, $4n$ modular exponentiations (small exponent), n modular exponentiations (large exponent) and $5n$ modular multiplications. In Step 4, Alice does n decryptions. Afterward, in Steps 5 and 6, Bob performs $4n$ modular exponentiations (large exponent), $2n$ modular multiplications, and n inverse tangents. Finally in Step 7, Alice does $2n$ decryptions and n inverse tangents. In the following, the computation cost per each party, as well as total computation cost, is calculated.

Alice's computation cost: $6nT_e + 3nT_d + nT_{atan2}$

Bob's computation cost: $T_e + 4nT_{se} + 5nT_{le} + 5nT_i + 7nT_m + nT_{atan2}$

Total computation cost: $(6n+1)T_e + 3nT_d + 4nT_{se} + 5nT_{le} + 7nT_m + 5nT_i + 2nT_{atan2}$

Protocol 2: In Step 3 of Protocol 2, Bob does 3 encryptions. Next, in Step 5, Alice conducts $2n$ encryptions, $4n$ modular exponentiations (small exponent) and $5n$ modular multiplications. Finally, in Step 6, Alice and Bob employ n times 1-out-of-p^2 Oblivious Transfer protocol. Moreover, Alice does a total of $2n(p-1)$ encryptions, meanwhile, Bob conducts a total of $2np$ decryptions and np^2 inverse tangents. Accordingly, the computation cost is calculated as follows.

Alice's computation cost: $2npT_e + 4nT_{se} + 2nT_i + 5nT_m + nT_{OT_A}$

Bob's computation cost: $3T_e + 2npT_d + np^2T_{atan2} + nT_{OT_B}$

Total computation cost: $(2np+3)T_e + 2npT_d + 4nT_{se} + 2nT_i + 5nT_m + np^2T_{atan2} + n(T_{OT_A} + T_{OT_B})$

5.2.2 Communication Overhead

In this section, we measure the communication cost of both protocols in bits. For simplicity, the communication cost of sending plaintext data is neglected. If N is modulus in Paillier's homomorphic encryption scheme, then a ciphertext has a maximum size of $2\log N$ bits. Moreover, the communication cost of invoking 1-out-of-p^2 Oblivious Transfer protocol is denoted by C_{OT}.

Protocol 1: In Step 1 of Protocol 1, Alice sends a total of $6n$ ciphertexts to Bob. Next, in Step 3 and 6, Bob sends a total of $3n$ ciphertexts back to Alice. Therefore, the overall communication cost is $18n\log N (= 9n \times 2\log N)$ bits.

Protocol 2: In Step 3 of Protocol 2, Bob sends 3 ciphertexts to Alice. Next,

in Step 6, Alice and Bob employ n times 1-out-of-p^2 Oblivious Transfer protocol. Meanwhile, Alice sends a total of $2pn$ ciphertexts to Bob. Therefore, the overall communication cost is $(4pn + 6)\log N (= (2pn + 3) \times 2\log N) + nC_{OT}$ bits.

5.2.3 Communication Round

Here, the round complexity of the two protocols is presented with respect to the number of communication needed to compute the result.

Protocol 1: Alice and Bob exchange messages four times.

Protocol 2: This protocol relies on the 1-out-of-p^2 Oblivious Transfer protocol that should be invoked n times to make the inclusion judgment. Thus, the round complexity of Protocol 2 is $O(n)$.

5.2.4 Implementation Results

In this section, we use the cost of cryptographic primitives presented in [MB16], in order to estimate the computation cost of our proposed protocols in a real implementation (see Table 3). As can be seen, the cost of encryption, decryption and modular large exponentiation are much higher compared to modular small exponentiation and modular multiplication. Moreover, as mentioned in [LT05], each inversion takes one modular multiplication. Therefore, for simplicity, we just consider the computation cost of encryption, decryption and modular large exponentiation in our estimation, and the cost of other operations (i.e., inversion, modular small exponentiation, modular multiplication and inverse tangent) are neglected. Let us set $\log N = 1024$ and $p = 10$. In the following, the costs of both protocols are estimated.

Table 3: Cost of cryptographic primitives (ms) [MB16]

Paillier cryptosystem	
Encryption	17.14
Decryption	15.69
Modular exponentiation (small exponent)	0.52
Modular exponentiation (large exponent)	16.15
Modular multiplication	0.017

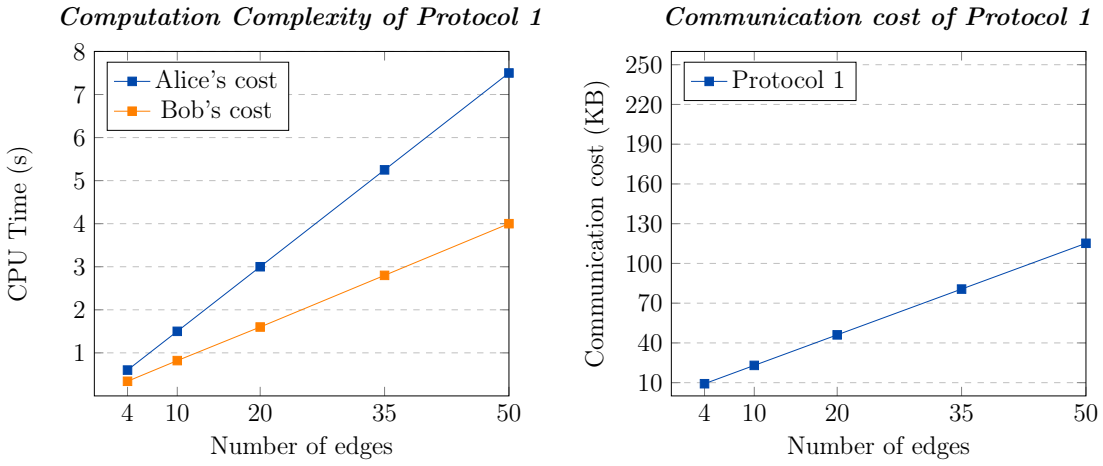


Figure 22: Computation complexity and communication overhead of Protocol 1.

Figure 22 shows the estimated CPU time required at the two parties, and the overall communication overhead in Protocol 1, when the number of edges varies. Both costs scale linearly with the number of edges in Alice's polygon. Moreover, Alice's CPU time increases more sharply than Bob's CPU time, since Alice needs to perform a lot of encryptions and decryptions to compute the result.

We compared our results with similar protocol proposed in [MB16] for a rectangular region[2]. In [MB16], the total computational cost is around 2 seconds, while in our protocol (Protocol 1) the total estimated time is about 0.95 seconds, increasing computational efficiency by up to 52.5%. Moreover, the rectangular region incurs a communication cost of 90 KB in [MB16], while the incurred communication cost in Protocol 1 is about 9.2 KB, almost reducing the communication complexity by up to 89.7%. Therefore, we have achieved a significant reduction of the communication overhead and the computation complexity in Protocol 1 compared to [MB16].

Figure 23 shows the approximate time and communication complexity of Protocol 2. Since 1-out-of-N oblivious transfer protocol has been extensively studied in various flavors and security models (cf. [Ste98, NP99, AIR01, NP01, Tze02, CO15]), the cost of this protocol would differ depending on the chosen scheme. In [Tze02], Tzeng gives an efficient 1-out-of-N OT that is secure in the random oracle model. In his proposed protocol, the receiver (Alice) needs to compute 2 modular exponentiations and the sender (Bob) computes $2N$ modular exponentiations. Meanwhile, a total of $2N + 1$ ciphertexts are exchanged between two parties.

[2]A rectangular region would probably be the most common area defined by Alice

Let us employ the 1-out-of-N OT proposed in [Tze02] in our estimation in order to estimate the complexity of the 1-out-of-p^2 oblivious transfer protocol in our solution. The cost of modular exponentiation in ElGamal cryptosystem presented in [MB16] is 1.38 ms. In our estimation, we set $p = 10$. Thus, if the number of edges is equal to fifty (i.e., $n = 50$), the probability that Bob might guess some information about Alice's polygon is much less in comparison with the case when the number of edges is equal to 4. However, setting $n = 50$ demands fifty times OT invocations that can be considered as the performance bottleneck regarding computation complexity and communication overhead.

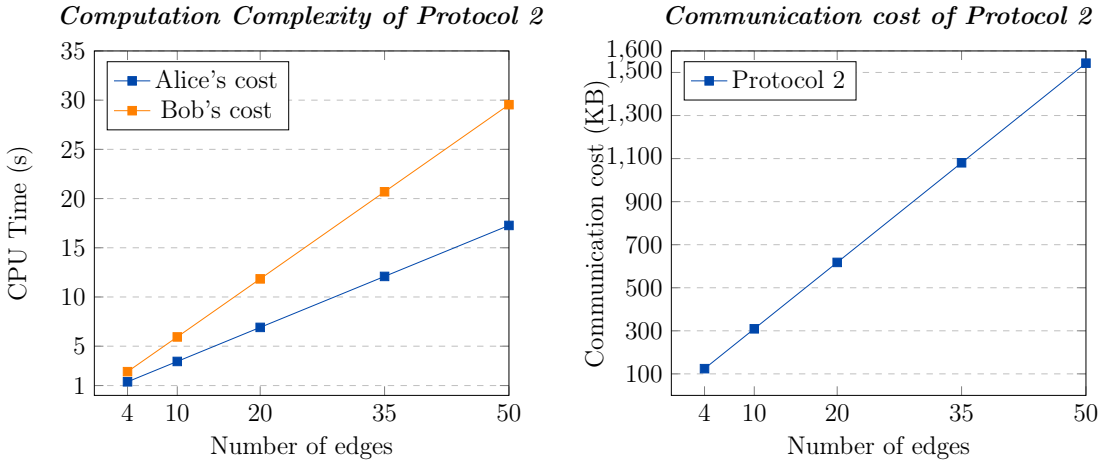


Figure 23: Computation complexity and communication overhead of Protocol 2.

By comparing the performance of both protocols, we can conclude that the communication overhead of Protocol 2 is about 13 times that of Protocol 1 that does not employ the oblivious transfer protocol. Moreover, the total computational cost of Protocol 2 is about 4 times that of Protocol 1. Therefore, Protocol 1 is more efficient than Protocol 2.

5.3 Privacy-Performance Trade-off

Figure 24 displays a graphical representation of the proposed protocols with respect to the privacy-performance trade-off achieved. As it is illustrated, in overall, Protocol 1 has better performance than Protocol 2. However, better efficiency is achieved by sacrificing the privacy of participants in relatively minor cases as discussed before. On the other hand, Protocol 2 provides stronger privacy protection for both parties during the protocol execution.

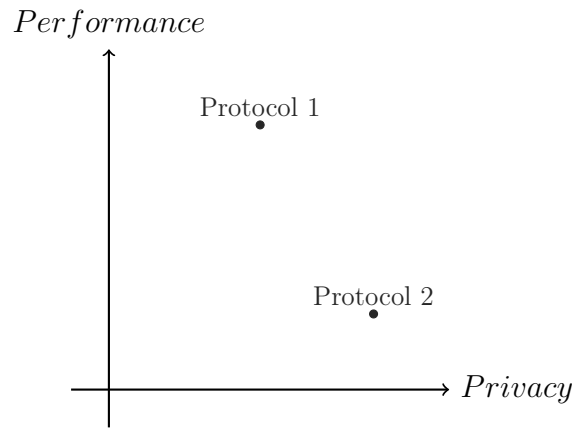


Figure 24: Privacy-Performance Trade-off

6 Conclusions and Future Work

In this thesis we investigated the point-in-polygon problem for complex polygonal domains; then we utilized the angle summation algorithm to address the private proximity detection for arbitrary concave or convex polygons. In this regard, we proposed two privacy-preserving protocols for the point-inclusion problem that allows a user to define an arbitrary geographic region on the map and check whether his/her friend is located therein. The proposed protocols are based on the secure two-party computation. It is noteworthy that extending our solution for use with more than two parties is straightforward and requires multiple executions of a protocol.

We also have analyzed the computation, round and communication complexity of both proposed protocols. The complexity analysis demonstrates that Protocol 1 has far better performance than Protocol 2. Since Protocol 2 employs the 1-out-of-N oblivious transfer protocol, the large number of OT invocations is considered as the performance bottleneck regarding computation complexity and communication overhead. However, Protocol 2 is more secure than Protocol 1, and provides stronger privacy protection for private inputs of two parties. It guarantees that no sensitive information regarding the location of two involved parties is disclosed to the other party. Moreover, in comparison with previous solutions, Protocol 1 reduces the number of communication rounds significantly.

Future Work

To the best of our knowledge, this is the first time that the proposed privacy-preserving protocols for the point-inclusion problem are applicable to all complex polygonal domains without partitioning a complex polygon into the number of convex pieces. As discussed in Chapter 5, there are some occasions that the privacy of users might be violated. For this reason, we believe that our proposed protocols could be further improved to provide more privacy. In the following, several issues that deserve further investigation are outlined.

In this thesis, we have assumed that the two parties are semi-honest. However, there could be malicious adversaries that try to manipulate the protocol result by sending corrupted data. Therefore, some advanced investigation approaches are needed to detect such adversaries and fake data.

From a technical point of view, some optimizations need a deeper investigation. Firstly, the utilized point-in-polygon algorithm can be refined to handle special cases, e.g., Bob might coincide with one of the vertices of P or may lie on one of P's edges. In [HA01], a detection algorithm for a vertex or an edge coincidence is suggested that might be applicable to our proposed privacy-preserving protocols. Secondly, Protocol 1 can be optimized to protect the signs of determinants against disclosure to Bob. As a result, protecting the privacy of Alice's polygon against a chosen-point attack will be accomplished. The *Secure Absolute Value Sub Protocol* proposed in [JTH12] might give some ideas for further improvements.

Another issue that should be addressed is that if Bob lies inside Alice's proximity area then some sensitive information might be divulged. The reason is that Alice could specify a very small area on the map, such as a mosque, church or a hospital, without Bob's approval. One possible solution suggested in [MB16] is to allow Bob to define a specific threshold aiming that he would not be found within an area smaller than the defined threshold. Therefore, Bob can supervise his location disclosure during the protocol execution. However, utilizing the aforementioned solution in our proposed protocols needs further investigation.

Another direction we are considering is a protection solution against *brute-force attack*. Our proposed privacy-preserving schemes cannot preserve the privacy of Bob's location against brute-force attack planned by Alice, i.e. she can initiate a series of proximity detection queries covering the whole area where Bob might be located. One proposed solution for Bob is to reject consecutive queries originated from Alice that are not sufficiently apart in time.

References

AD01 Atallah, M. J. and Du, W., Secure multi-party computational geometry. *Workshop on Algorithms and Data Structures*. Springer, 2001, pages 165–179, URL https://doi.org/10.1007/3-540-44634-6_16.

AIR01 Aiello, B., Ishai, Y. and Reingold, O., Priced oblivious transfer: How to sell digital goods. *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2001, pages 119–135, URL https://doi.org/10.1007/3-540-44987-6_8.

BCR86 Brassard, G., Crépeau, C. and Robert, J.-M., All-or-nothing disclosure of secrets. *Conference on the Theory and Application of Cryptographic Techniques*. Springer, 1986, pages 234–238, URL https://doi.org/10.1007/3-540-47721-7_17.

BJMR75 Berg, G., Julian, W., Mines, R. and Richman, F., The constructive Jordan curve theorem. *The Rocky Mountain Journal of Mathematics*, pages 225–236. URL https://www.jstor.org/stable/44236427.

BN11 Brunton, F. and Nissenbaum, H., Vernacular resistance to data collection and analysis: A political theory of obfuscation. *First Monday*, 16,5(2011).

CDN01 Cramer, R., Damgård, I. and Nielsen, J. B., Multiparty computation from threshold homomorphic encryption. *In EUROCRYPT*. Springer, 2001, pages 280–300, URL https://doi.org/10.1007/3-540-44987-6_18.

CGS97 Cramer, R., Gennaro, R. and Schoenmakers, B., A secure and optimally efficient multi-authority election scheme. *European transactions on Telecommunications*, 8,5(1997), pages 481–490.

CKGS98 Chor, B., Kushilevitz, E., Goldreich, O. and Sudan, M., Private information retrieval. *J. ACM*, 45,6(1998), pages 965–981. URL http://doi.acm.org/10.1145/293347.293350.

CML06 Chow, C.-Y., Mokbel, M. F. and Liu, X., A peer-to-peer spatial cloaking algorithm for anonymous location-based service. *Proceedings of the 14th annual ACM international symposium on Advances in geographic*

information systems. ACM, 2006, pages 171–178, URL https://doi.org/10.1145/1183471.1183500.

CML11 Chow, C.-Y., Mokbel, M. F. and Liu, X., Spatial cloaking for anonymous location-based services in mobile peer-to-peer environments. *GeoInformatica*, 15,2(2011), pages 351–380. URL https://doi.org/10.1007/s10707-009-0099-y.

CO15 Chou, T. and Orlandi, C., The simplest protocol for oblivious transfer. *International Conference on Cryptology and Information Security in Latin America*. Springer, 2015, pages 40–58, URL https://doi.org/10.1007/978-3-319-22174-8_3.

Cor08 Corral, M., *Vector Calculus*. Michael Corral, 2008.

DA01a Du, W. and Atallah, M. J., Privacy-preserving cooperative statistical analysis. *Seventeenth Annual Computer Security Applications Conference*. IEEE, 2001, pages 102–110, URL https://doi.org/10.1109/ACSAC.2001.991526.

DA01b Du, W. and Atallah, M. J., Secure multi-party computation problems and their applications: a review and open problems. *in Proceedings of the 2001 Workshop on New Security Paradigms*. ACM, 2001, pages 13–22, URL https://surface.syr.edu/eecs/11.

DBVKOS97 De Berg, M., Van Kreveld, M., Overmars, M. and Schwarzkopf, O., *Computational Geometry*. Springer, 1997.

DG10 Danezis, G. and Gürses, S., A critical review of 10 years of privacy technology. *Proceedings of surveillance cultures: a global surveillance society*, 2010, pages 1–16.

DGK07 Damgård, I., Geisler, M. and Krøigaard, M., Efficient and secure comparison for on-line auctions. *Australasian Conference on Information Security and Privacy*. Springer, 2007, pages 416–430, URL https://dl.acm.org/citation.cfm?id=1770231.1770269.

DMS04 Dingledine, R., Mathewson, N. and Syverson, P., Tor: The second-generation onion router. *In Proceedings of the 13th USENIX Security Symposium*. USENIX, 2004, pages 303–320.

EGL85 Even, S., Goldreich, O. and Lempel, A., A randomized protocol for signing contracts. *Communications of the ACM*, 28,6(1985), pages 637–647.

Fra11 Franz, M., *Secure Computations on Non-integer Values*. Ph.D. thesis, Technische Universität Darmstadt, 2011.

Ghi13 Ghinita, G., Privacy for location-based services. *Synthesis Lectures on Information Security, Privacy, and Trust*, volume 4. Morgan & Claypool Publishers, 2013, pages 1–85, URL `http://books.google.com/books?id=RB-fcnlloxoC`.

GKK+08 Ghinita, G., Kalnis, P., Khoshgozaran, A., Shahabi, C. and Tan, K.-L., Private queries in location based services: anonymizers are not necessary. *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*. ACM, 2008, pages 121–132, URL `https://doi.org/10.1145/1376616.1376631`.

GKKB09 Ghinita, G., Kalnis, P., Kantarcioglu, M. and Bertino, E., A hybrid technique for private location-based queries with database protection. *International Symposium on Spatial and Temporal Databases*. Springer, 2009, pages 98–116, URL `https://doi.org/10.1007/978-3-642-02982-0_9`.

GL08 Gedik, B. and Liu, L., Protecting location privacy with personalized k-anonymity: Architecture and algorithms. *IEEE Transactions on Mobile Computing*, 7,1(2008), pages 1–18. URL `https://doi.org/10.1109/TMC.2007.1062`.

GMW87 Goldreich, O., Micali, S. and Wigderson, A., How to play any mental game. *Proceedings of the nineteenth annual ACM symposium on Theory of computing*. ACM, 1987, pages 218–229, URL `https://doi.org/10.1145/28395.28420`.

Gol98 Goldreich, O., Secure multi-party computation. *Manuscript. Preliminary version*, 78. URL `http://www.wisdom.weizmann.ac.il/~oded/pp.html`.

Gol09 Goldreich, O., *Foundations of cryptography: volume 2, basic applications*. Cambridge university press, 2009.

HA01 Hormann, K. and Agathos, A., The point in polygon problem for arbitrary polygons. *Computational Geometry. Theory and Applications*, 20,3(2001), pages 131–144. URL https://doi.org/10.1016/S0925-7721(01)00012-8.

Her16 Herrmann, M., *Privacy in Location-Based Services*. Ph.D. thesis, Leuven, Belgium: KU Leuven–Faculty of Engineering Science, 2016.

HGXA06 Hoh, B., Gruteser, M., Xiong, H. and Alrabady, A., Enhancing security and privacy in traffic-monitoring systems. *IEEE Pervasive Computing*, 5,4(2006), pages 38–46. URL https://doi.org/10.1109/MPRV.2006.69.

JKS+18 Järvinen, K., Kiss, Á., Schneider, T., Tkachenko, O. and Yang, Z., Faster privacy-preserving location proximity schemes. *International Conference on Cryptology and Network Security*. Springer, 2018, pages 3–22.

JTH12 Jeckmans, A., Tang, Q. and Hartel, P., Privacy-preserving collaborative filtering based on horizontally partitioned dataset. *2012 International Conference on Collaboration Technologies and Systems (CTS)*. IEEE, 2012, pages 439–446, URL https://doi.org/10.1109/CTS.2012.6261088.

KGMP07 Kalnis, P., Ghinita, G., Mouratidis, K. and Papadias, D., Preventing location-based identity inference in anonymous spatial queries. *IEEE transactions on knowledge and data engineering*, 19,12(2007), pages 1719–1733. URL https://doi.org/10.1109/TKDE.2007.190662.

KMVOV96 Katz, J., Menezes, A. J., Van Oorschot, P. C. and Vanstone, S. A., *Handbook of applied cryptography*. CRC press, 1996.

KO97 Kushilevitz, E. and Ostrovsky, R., Replication is not needed: Single database, computationally-private information retrieval. *Proceedings 38th Annual Symposium on Foundations of Computer Science*. IEEE, 1997, pages 364–373, URL https://doi.org/10.1109/SFCS.1997.646125.

Küp05 Küpper, A., *Location-based services: fundamentals and operation*. New York: Wiley, 2005.

LD05 Li, S.-D. and Dai, Y.-Q., Secure two-party computational geometry. *Journal of Computer Science and Technology*, 20,2(2005), pages 258–263. URL https://doi.org/10.1007/s11390-005-0258-z.

LHZ07 Luo, Y.-L., Huang, L.-S. and Zhong, H., Secure two-party point-circle inclusion problem. *Journal of Computer Science and Technology*, 22,1(2007), pages 88–91. URL https://doi.org/10.1007/s11390-007-9011-0.

LL09 Lipschutz, S. and Lipson, M., *Linear Algebra: Schaum's Outlines*. McGraw-Hill, 2009.

LT05 Lin, H.-Y. and Tzeng, W.-G., An efficient solution to the millionaires' problem based on homomorphic encryption. *International Conference on Applied Cryptography and Network Security*. Springer, 2005, pages 456–466, URL https://doi.org/10.1007/11496137_31.

MB16 Mu, B. and Bakiras, S., Private proximity detection for convex polygons. *Tsinghua Science and Technology*, 21,3(2016), pages 270–280. URL https://doi.org/10.1109/TST.2016.7488738.

MBF09 Mascetti, S., Bettini, C. and Freni, D., Longitude: Centralized privacy-preserving computation of users' proximity. *Workshop on Secure Data Management*. Springer, 2009, pages 142–157, URL https://doi.org/10.1007/978-3-642-04219-5_9.

NP99 Naor, M. and Pinkas, B., Oblivious transfer and polynomial evaluation. *Proceedings of the thirty-first annual ACM symposium on Theory of computing*. ACM, 1999, pages 245–254.

NP01 Naor, M. and Pinkas, B., Efficient oblivious transfer protocols. *Proceedings of the twelfth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, 2001, pages 448–457, URL http://dl.acm.org/citation.cfm?id=365411.365502.

NTL+11 Narayanan, A., Thiagarajan, N., Lakhani, M., Hamburg, M., Boneh, D. et al., Location privacy via private proximity testing. *NDSS*, volume 11, 2011.

Pai99 Paillier, P., Public-key cryptosystems based on composite degree residuosity classes. *In EUROCRYPT*. Springer, 1999, pages 223–238, URL https://doi.org/10.1007/3-540-48910-X_16.

pri02 Stanford encyclopedia of philosophy. privacy. URL https://plato.stanford.edu/entries/privacy/.

PS88 Preparata, F. P. and Shamos, M. I., *Computational Geometry: an Introduction*. Springer, 1988.

ŠG14 Šeděnka, J. and Gasti, P., Privacy-preserving distance computation and proximity testing on earth, done right. *Proceedings of the 9th ACM symposium on Information, computer and communications security*. ACM, 2014, pages 99–110, URL https://doi.org/10.1145/2590296.2590307.

Sha49 Shannon, C. E., Communication theory of secrecy systems. *Bell system technical journal*, 28,4(1949), pages 656–715. URL https://doi.org/10.1002/j.1538-7305.1949.tb00928.x.

SS05 Sang, Y. and Shen, H., A scheme for testing privacy state in pervasive sensor networks. *19th International Conference on Advanced Information Networking and Applications (AINA'05) Volume 1 (AINA papers)*, volume 2. IEEE, 2005, pages 644–648, URL https://doi.org/10.1109/AINA.2005.60.

Ste98 Stern, J. P., A new and efficient all-or-nothing disclosure of secrets protocol. *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 1998, pages 357–371, URL https://doi.org/10.1007/3-540-49649-1_28.

ŠTŠ+09 Šikšnys, L., Thomsen, J. R., Šaltenis, S., Yiu, M. L. and Andersen, O., A location privacy aware friend locator. *International Symposium on Spatial and Temporal Databases*. Springer, 2009, pages 405–410, URL https://doi.org/10.1007/978-3-642-02982-0_29.

ŠTŠY10 Šikšnys, L., Thomsen, J. R., Šaltenis, S. and Yiu, M. L., Private and flexible proximity detection in mobile social networks. *2010 Eleventh International Conference on Mobile Data Management*. IEEE, 2010, pages 75–84, URL https://doi.org/10.1109/MDM.2010.43.

Swe02 Sweeney, L., k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10,05(2002), pages 557–570. URL https://doi.org/10.1142/S0218488502001648.

Tho09 Thomas, T., Secure two-party protocols for point inclusion problem. *International Journal of Network Security*, 9, pages 1–7.

Tze02 Tzeng, W.-G., Efficient 1-out-n oblivious transfer schemes. *International Workshop on Public Key Cryptography*. Springer, 2002, pages 159–171, URL https://doi.org/10.1007/3-540-45664-3_11.

USD11 Ukil, A., Shah, V. H. and Deck, B., Fast computation of arctangent functions for embedded applications: A comparative analysis. *2011 IEEE International Symposium on Industrial Electronics*. IEEE, 2011, pages 1206–1211, URL https://doi.org/10.1109/ISIE.2011.5984330.

Wei94 Weiler, K., Graphics gems iv. Academic Press Professional, Inc., San Diego, CA, USA, 1994, chapter An Incremental Angle Point in Polygon Test, pages 16–23, URL http://dl.acm.org/citation.cfm?id=180895.180898.

Wes67 Westin, A. F., *Privacy and freedom*. New York: Atheneum Press, 1967. URL https://doi.org/10.1093/sw/13.4.114-a.

WW10 Wang, S. and Wang, X. S., In-device spatial cloaking for mobile user privacy assisted by the cloud. *2010 Eleventh International Conference on Mobile Data Management*. IEEE, 2010, pages 381–386, URL https://doi.org/10.1109/MDM.2010.82.

Yao82 Yao, A. C.-C., Protocols for secure computations. *23rd Annual Symposium on Foundations of Computer Science*. URL https://doi.org/10.1109/SFCS.1982.38.

YLWY10 Yun, Y., Liusheng, H., Wei, Y. and Youwen, Z., Efficient protocols for point-convex hull inclusion decision problems. *Journal of Networks*, 5,5(2010), page 559.

YSZ11 Yang, B., Sun, A. and Zhang, W., Secure two-party protocols on planar circles. *Journal of Information & Computational Science*, 8,1(2011), pages 29–40.

YSZ12 Yang, B., Shao, Z. and Zhang, W., Secure two-party protocols on planar convex hulls. *Journal of Information & Computational Science*, 9,4(2012), pages 915–929.

ZGH07 Zhong, G., Goldberg, I. and Hengartner, U., Louis, Lester and Pierre: Three protocols for location privacy. *International Workshop on Privacy Enhancing Technologies*. Springer, 2007, pages 62–76, URL `https://doi.org/10.1007/978-3-540-75551-7_5`.

ZWL+18 Zhu, H., Wang, F., Lu, R., Liu, F., Fu, G. and Li, H., Efficient and privacy-preserving proximity detection schemes for social applications. *IEEE Internet of Things Journal*, 5,4(2018), pages 2947–2957. URL `https://doi.org/10.1109/JIOT.2017.2766701`.